AN EXPERT SYSTEM FOR DIAGNOSIS OF FAULTS USING FAULT TREES

By

AMITAVA GUPTA

NUCLEAR ENGINEERING AND TECHNOLOGY PROGRAM

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

DECEMBER , 1991

AN EXPERT SYSTEM FOR DIAGNOSIS OF FAULTS USING FAULT TREES

A Thesis Submitted

in Partial Fulfilment of the Requirements

for the Degree of

MASTER OF TECHNOLOGY

By

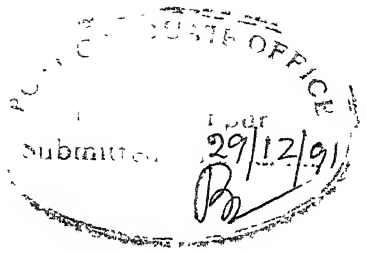
AMITAVA GUPTA

to the

NUCLEAR ENGINEERING AND TECHNOLOGY PROGRAM

INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

DECEMBER, 1991



CERTIFICATE

It is certified that the work contained in this thesis entitled 'An Expert system for diagnosis of Faults using Fault Trees' has been carried out under our supervision and that the work has not been submitted elsewhere for a degree.

K. Sri Ram

Dr. K. Sri Ram

Professor, N.E.T Program

I.I.T. Kanpur.

Karnick

Dr. H. Karnick

Asst. Professor,

Dept. of C.S.& E.

I.I.T. Kanpur.

December, 1991

NETP-1991-M-GUP-EXP

ACKNOWLEDGEMENTS

I am profoundly indebted to my thesis supervisors Dr.H.Karnick and Dr.K.Sri Ram.They have helped me at every stage of the work with unerring guidance and I have greatly enjoyed their company.

I would like to thank Dr.M.S.Kalra and Dr.A.Sen Gupta for the help I have received from them.I would like to thank Dr.P.Munshi specially, for the help and encouragement I have received from him throughout my entire stay.I would also like to thank Mr.V.S.Tomar for the patience with which he has solved the infinite many confusions I requested him to look into.

Mr.S.S.Pathak, Mr.S.Pandimani,Mr.Manmohan Pandey and Mr.K.M.Singh helped me a lot in computer related matters.Mr.A.C.Trivedi, Engineer,PC maintenance cell, has provided me with relevant information for PC fault diagnosis.

All my friends have given me a nice time throughout the entire period of my stay.

AMITAVA GUPTA

CONTENTS

LIST OF TABLES

LIST OF SELECTED SYMBOLS

NOTATIONS

ρ	reactivity
λ	decay rate for delayed neutrons precursors
β	fraction of delayed neutrons
s	complex frequency
e_i	i^{th} basic event(fault)
s_i	i^{th} intermediate event(symptom)
T	TOP event
E	Set of all basic events
E_a	Set of all basic events which have not occurred
E_d	Set of basic events which are definitely known to have occurred.
E_x	Set of basic events about which nothing definite is known
S	Set of all intermediate events
S_a	Set of all intermediate events having instrumented fault indicators
S_x	Set of intermediate events about which nothing definite can be concluded.
λ_i	Equivalent failure rate associated with basic event e_i

ABBREVIATIONS

CANDU	Canadian Deuterium Uranium reactor
EMSR	Electromechanical Shutdown Rod
LPR	Liquid Poison Rod
IBM	International Business Machines

ABSTRACT

Diagnosis, using Expert systems, has become extremely popular nowadays. A variety of methods have been proposed by many researchers to develop intelligent diagnosis techniques using knowledge about the structure and behavior of a system. One of the many ways to represent this knowledge is the 'Fault Tree' representation. Though mostly used for reliability assessment, the fault trees represent the cause effect relationship between the symptoms and the causes in a structured manner and at the same time provide data for probabilistic analyses.

For most practical systems, where a set of symptoms point to a large number of faults, an uncertainty exists about the occurrence of a particular fault. Diagnosis, using fault trees, attempts to use the logical formulation representing the symptom - cause relationship and couples it with probabilistic techniques to diagnose a fault, when an uncertainty exists about its occurrence.

In the present case, the fault tree has been constructed for the Electromechanical Shutdown Rod system of CANDU reactors and for some selected problems associated with the IBM PC and the search strategy has been tested using these fault trees. The results show that in most cases where occurrence of faults is guided by probability of occurrence of that fault, probabilistic techniques can reduce the number of searches required to diagnose a fault(when some uncertainty exists about its occurrence) greatly from what would have been required by an exhaustive approach.

For testing the effectiveness of the diagnosis strategy in the case of faults associated with the electromechanical shutdown rod system of CANDU reactors, a simulated version of the system was used. The

simulator was developed using properties of signal flow graphs. The tests for PC fault diagnosis have been carried out on an IBM compatible PC/XT having known problems. An interface has also been developed to enable the user to develop the fault tree in an interactive manner.

CHAPTER ONE

INTRODUCTION

Any functional item will have a certain desired or required performance specification and a failure is said to have occurred if it fails to operate in the desired manner. This project attempts to diagnose faults using failure rates or reliability data and uses a MYCIN[3] like uncertainty handling technique.

A program has been developed to simulate faults for electromechanical systems and it can simulate faults which produce changes in the steady state behavior of such systems. The EMSR (Electromechanical Shutdown Rod) system for CANDU reactors has been simulated with it and its behavior has been simulated under normal and faulted conditions using the above program and the system response has been used in fault diagnosis using the reliability data available for the CANDU EMSR system through the diagnosis software. The diagnosis strategy has also been applied to some selected problems associated with IBM personal computers.

1.1 FAULT AND FAULT TYPES

Failure modes for most systems, electronic systems in particular can be classified into two broad categories[2]:-

(a) **Through drift failures:-** Due to gradual degradation faults occur when a certain threshold value is crossed. A typical example is the longevity problem of a rechargable battery, where the internal resistance increases in direct proportion to the time the load is

applied and recovers at a rate equal to one half of this value. For such a system, a failure is said to have occurred when the internal resistance change crosses a value of 4.00 ohms.

(b) **Chance failure or catastrophic failure:-** Such a failure occurs at random within the operational time of an equipment and before wear becomes pre-dominant. Typical examples are shorted electron tubes and wire breakages. In this case, the lifetime of the component may be represented by a random variable T, for which the survival function $\nu(t)$[1] is defined as:-

$$\nu(t) = p_r(T>t)$$

where $p(T>t)$ is the probability that a component has a failure at a time after t.

If the above hypothesis is accepted, then the probability that a component has a failure at a time earlier than t is:-

$$p_r(t)(T \leq t) = 1 - \nu(t) = \phi(t)$$

$\phi(t)$ is called the distribution function and $\nu(t)$ is called the complementary distribution function.

Survival functions are broadly classified into three types as follows[1]:-

(a) **Type I.** the random variable T representing the lifetime of a component is a discrete variable. The survival function is a step function.

(b) **Type IIa.** The random variable T takes its values in $[\alpha, 0)$ and its distribution function $\phi(t)$ and survival function $\nu(t)$ are piece wise continuous for all t and $\phi(t)$ and $\nu(t)$ admit a derivative in

the interval where they are defined.

(c)**Type IIb.** The random variable T takes its values in $[\alpha,0]$ and the corresponding distribution function $\Phi(t)$ and the survival function $\nu(t)$ are piece wise continuous and make at least one jump.

The survival Law defined by a Rate Deterioration is applicable in case **IIa** and is mostly used.

1.2 DIAGNOSIS: DETERMINISTIC & NON DETERMINISTIC MODELS

(a)**Deterministic model**

Let $S \equiv [s_1, s_2, s_3, \ldots\ldots s_n]$ represent a finite set of symptoms.

Let $F \equiv [f_1, f_2, f_3, \ldots\ldots f_m]$ represent a finite set of faults.

If each of the faults $f_1, f_2, f_3, f_4, \ldots\ldots f_m$ produce sets of observable $sf_1, sf_2, sf_3, \ldots\ldots sf_m$ where each $sf_i \in S$, and there exist a mapping function $Q(f)$ for each f_i which maps the corresponding sf_i onto F, then each of the faults can be uniquely determined and the model is deterministic.

(b)**Non-deterministic model**

All information needed to diagnose a fault is rarely available. A variety of techniques for handling incomplete information within computer programs have been proposed. Some examples of such reasoning techniques[4] are as follows:-

(i)**Non-monotonic reasoning:** Here conclusions are defusible and so can be deleted in the on-going reasoning to prove or disprove a particular diagnosis.

(ii)**Probabilistic reasoning:** This technique is used to represent likely but uncertain inferences.

(iii)**Techniques using concept of belief spaces** which allow representation of nested models of sets of beliefs.

1.3 REASONING BASED DIAGNOSIS TECHNIQUES.

Most reasoning based diagnosis techniques use the Fault-tree method. The fault-tree reconstructs the fault from the failure of basic components using binary logic. The essential steps involved in a fault tree construction[5] are:-

(i)**Definition of the TOP event.** According to standard fault-tree terminology, the TOP event is the event whose probability of occurrence is desired. The definition of TOP event is in terms of system hardware. For diagnosis purposes, the TOP event is the root of the AND-OR tree representing the fault-tree.

(ii)**Definition of a set of INTERMEDIATE EVENTS.** According to fault-tree terminology, INTERMEDIATE EVENTs are those events which can cause TOP event either individually, or in some mixed mode. Their relation to the TOP event is defined by boolean operators AND and OR. For diagnosis purposes, these are used as the symptoms.

(iii)**Definition of a set of BASIC EVENTs.** Each intermediate event is to be treated as TOP events and step(ii) is repeated till a set of BASIC EVENTs is obtained. These are events defining failures of such components which needs no further development for analysis purposes. Basic events are the leaves of the fault-tree and

reliability or failure data for such events is either available or it has to be collected for these nodes.

(iv)**Editing the Fault-tree** so constructed to avoid omissions,logical discrepancies and repetitions.

It is to be noted that fault-trees for components which fail in more than one mode are separately constructed and substituted at appropriate positions in the system fault-tree.

The fault-tree method is a versatile method for analyzing complex systems.Some of the advantages of fault-tree analysis are:-

(a)Directing the analyst to diagnose failures in a deductive way.

(b)Pointing out the important aspects for the failure of interest.

(c)Providing options for qualitative or quantitative system reliability analysis.

(d)Allowing the analyst to concentrate on a particular system at a time.

Table. 1.1 is a representation of a few noteworthy programs developed in the last twenty years for fault-tree analysis.They have been categorized into a few groups.Group one consists of fault-tree construction programs.Groups two,three and four are the analysis programs.The analysis programs may be divided into two general types viz.those which directly produce numerical results(group four) and those which first qualitatively and then quantitatively(group three) analyze the logic system.

Table 1.1

A few noteworthy computer programs related to fault-trees

(1) Fault Tree Construction programs

Draft(Fuss ell,1973)

(2) Programs finding minimal cut sets[7]

Prep[Vesely,1970]

(3) Numerical evaluation programs

KITT1,KITT2(Vesely,1974)

(4) Direct Evaluation programs

SAMPLE(Wash 1400)

1.4 SURVEY OF EXISTING LITERATURE.

Diagnosis as a problem has been dealt with in a number of ways and in a wide variety of realms ranging from medical diagnosis to trouble-shooting in personal computers[11].A generalized approach has been put forward by Raymond Reiter,Johan de Kleer and Daniel G.Bobrow[10].In the method of diagnosis suggested by them the system has to be described in a suitable logic(preferably First Order) and uses nonmonotonic reasoning techniques.The method accommodates diagnostic reasoning in a wide variety of practical settings,including digital and analog circuits,medicine and database updates.

Some useful diagnosis techniques are available in the field of medical diagnosis.One such program is the MYCIN[3] using inexact

reasoning in the domain of uncertain knowledge. Another example is the TEIRESIAS[4] which provides a way for doctors to interact with MYCIN.

Diagnosis has also been attempted using system description in a variety of ways. Accordingly, a large number of programs have been developed which are dedicate to representing the system description. For example DRAFT[6] is a fault-tree construction program which proposes the component failure transfer function approach. The component failure transfer functions are obtained from a system independent analysis of every component appearing in the system for which the fault-tree is to be constructed. However, most of the programs associated with fault-trees deal with quantification of the fault tree for reliability estimation purposes and they are mostly based on Vesely's Kinetic Tree Theory. For diagnosis purposes, the fault-tree is treated as an AND-OR tree and can be used for diagnosis purposes. One such attempt has been made by R. Davis[8] where a diagnostic method using structure and behavior has been put forward and J de Kleer[9] in an attempt to diagnose multiple faults.

1.5. AN OVERVIEW OF THE DIAGNOSIS METHOD PROPOSED

In the present case, diagnosis is done by traversing the fault-tree of the system which is an AND-OR search tree starting at the TOP event. The search is a three tier process. Using deductive reasoning, the set of faults which have definitely occurred and the set of faults which have definitely not occurred are first identified. The remaining sub-set of the set of all possible faults

constitutes the set of faults about which uncertainty exists. The uncertainty is resolved using probabilistic techniques. For this purpose, probability of occurrence of a particular fault is computed using reliability data and the technique used is a modification of the medical diagnosis program MYCIN. The history of occurrence of a particular fault is also considered for calculating individual fault probabilities. A set of most probable faults is thus identified.

The third step in the traversal is based on user interrogations. A guided search is made over the set of most probable faults using deductions made earlier and also in the course of this search to determine which of the faults have actually occurred. This step is also repeated over the set of less probable faults if required.

Fault-trees were constructed for the EMSR(Electromechanical Shutdown Rod) system of the CANDU reactors and also for some aspects of the IBM personal Computer viz. problems related to Start up, Reading/Writing, Keyboard problems and the search strategy has been tested with these.

CHAPTER TWO

SIMULATION OF FAULTS

The simulator has been developed to study the steady-state behavior of faulty system and use such behavior for testing the diagnosis algorithm. It uses concept of transfer function and makes quasi steady state approximations to obtain values of outputs at intermediate time intervals. The formulation is as follows.

2.1 QUASI STEADY STATE APPROXIMATIONS FOR DYNAMIC INPUTS AND THEIR VALIDITY.

If a system can be described by a transfer function of the form

$$G(s) = \frac{a_n s^n + a_{n-1} s^{n-1} + a_{n-2} s^{n-2} + \quad + a_0}{b_m s^m + b_{m-1} s^{m-1} + b_{m-2} s^{m-2} + \quad + b_0}$$

where $a_o \neq 0, b_o \neq 0$, then the output of such a system $Y(t)|_{t=\alpha}$ at steady state due to an input $X(t)$ will be as follows:-

Case(i)

$X(t) = AU_{-1}(t)$ (step of height A)

$Y(t) = L^{-1}(A/s).G(s)$ and

$$Y(t)\big|_{t=\alpha} = \lim_{s \to 0} s.\ (A/s).G(s) = A.(a_o/b_o)$$

Case(ii)

$X(t) = AU_{-2}(t)$ i.e a ramp of slope A

$$Y(t) = L^{-1}(A/s^2).G(s).$$

$$Y(t)\big|_{t=\alpha} = \lim_{s \to 0} s.(A/s^{2}).G(s) = \alpha$$

case(iii)

$X(t) = AU_{-3}(t)$

$$Y(t) = L^{-1}(2.A/s^3).G(s)$$

$$Y(t)\big|_{t=\alpha} = \lim_{s \to 0} s.(2.A/s^{3}).G(s) = \alpha$$

It is thus clearly seen that the steady state value at t = α for ramp and parabolic inputs to the above system is infinity but the output has a finite value at any intermediate time instant and these values are computable using inverse Laplace transforms. Since inversion from 's' domain to 't' domain is a complicated process, a few approximations are to be made in cases (ii) and (iii) which are as follows:-

case(i)

$X(t) = AU_{-2}(t)$

$X(s) = A/s$

$Y(s) = (A/s).G(s) = A(P_1/s + P_2(s)/(b_m s^m + b_{m-1} s^{m-1} + \quad + b_0))..(i)$

where P_1 is a constant and $P_2(s)$ is any polynomial in s.
The term $P_2(s)$ may be broken down further into many more partial fractions and its inverse will produce the transient part of Y(t).
Concentrating on equation(i) and equating the coefficients of the corresponding powers of 's',we have,

$$P_1(b_m s^m + b_{m-1} s^{m-1} + \quad b_0) = A(a_n s^n + a_{n-1} s^{n-1} + \quad + a_0)$$

Equating powers of 's' from both the sides,we have,

$P_1 b_0 = A a_0$

or,$P_1 = A(a_0/b_0)$

or,$Y(s) = A(a_0/b_0).(1/s) + P_2(s)/(b_m s^m + b_{m-1} s^{m-1} + \quad b_0)$

After the initial phase,when transients die down $Y(t) = A(a_0/b_0)$ describes the behavior of the output.

case(ii)

$X(t) = AU_{-2}(t)$

$X(s) = A/s^2$

$Y(s) = (A/s^2).G(s)$

or, $Y(s) = P_1/s + P_2/s^2 + \quad P_s(s)/(b_m s^m + b_{m-1} s^{m-1} + \quad + b_o)$

Following similar analysis(as in the previous case) we have,

$P_1 b_o = A a_o$

or, $P_1 = A(a_o/b_o)$

again, $P_1 b_o + P_2 b_1 = A a_1$
or, $P_1 = A.(a_1 b_o - a_o b_1)/b^2_o$

After the initial transients die out, the output Y(t) can be described by the equation $Y(t) = (a_1 b_o - a_o b_1)/b^2_o + (a_o/b_o).t$

It can be seen clearly that in this case, there is a constant error(difference between the actual and approximated outputs) and this is equal in magnitude to P_1 if Y(t) is approximated by the relation $Y(t) = P_2.t$. The ratio $P_1/P_2.t$, therefore, defines the ratio of error to the approximated outputs for a ramp input to the aforesaid system. It is clearly seen that as the value of 't' increases, this ratio decreases.

case(iii)

$X(t) = AU_{-2}(t)$

$X(s) = 2.A/s^3$

$Y(s)=X(s).G(s) = P_1 + P_2/s + P_3/s^2 + P_4(s)/(b_m s^m + b_{m-1} s^{m-1} + \ldots + b_0)$

where P_1, P_2 and P_3 are constants and $P_4(s)$ is a polynomial in s.

Following similar analysis,as in the previous two cases,we have,

$P_1 = (2Aa_2 b_0^2 - 2Aa_0 b_2 b_0 - 2Aa_0 b_1^2 + 2Aa_1 b_1 b_0)/(b_0)^3$

$P_2 = 2A.(a_0 b_1 - a_1 b_0)/(b_0)^2$

$P = 2A.(a_0/b_0)$

Thus after the initial transients die out,the output can be described by the equation $Y(t) = 2A(a_0/b_0)t^2$ with an error equal to $P_1 + P_2 t$. It is evident,that unlike the previous case,the error does not remain constant but varies linearly with time. But the output being a function of t^2 and the error being a linear function of time,the ratio of error to the approximated output decreases with time.

From the above discussions it follows that for any general input of the form $X(t) = AU_{-n}(t)$, the output can be approximated by the relation $Y(t) = (n-1)!.(a_o/b_o)X(t)$ (a_o and b_o has already been defined) with an error which depends on the type of the input. Moreover, the transient parts being neglected in all cases, the approximations are valid for all $t \geq t_s$, where t_s is the settling time for the system. But since simulation for normal and faulted states is done

in the same frame of reference using the same set of approximations there will be sufficient qualitative difference between the normal and faulted states to differentiate between the two states.

2.2. SIMULATION BASED ON SIGNAL FLOW GRAPH MODEL.

For simulating systems with signal flow graphs, we use the transmission, addition and multiplication properties of signal flow graphs. The numbering of the nodes follows standard convention. Thus, if n_i and n_j are two nodes in the signal flow graph representing two variables x_i, x_j respectively (x_i, x_j may be functions of time, complex frequency or any other quantity), and if they are linked by a branch L directed from n_i to n_j, then the link or branch transmittance for the link L will be denoted by A_{ij} The input node is always designated as n_1.

Using the transmission property and the multiplication properties of signal flow graphs, a cascaded series connection of (n - 1) branches with transmittances described by $A_{12}, A_{23}, \ldots\ldots\ldots A_{(n-1)n}$ can be replaced by a single branch of transmission function equal to the product of the original (n - 1) branches and

$X_n = A_{12} \cdot A_{23} \cdot \ldots \cdot A_{(n-1)n} X_1$ where X_1 is the variable associated at node 1 and X_n is the variable associated with node n. This property holds good for any two nodes n and m with only series cascaded branches between them.

However, when feedback and feed forward branches are also present, the series multiplication property cannot be used directly. Feed forward branches are described by transmission functions of the form $A_{ij}|j > i$ and $j \neq i + 1$, and feedback paths are described by transmission functions of the form $A_{ij}|i>j$. It is to be noted that a branch connecting any two consecutive nodes i and i + 1 is a feed forward path, but we restrict our discussions to the classical definition of feed-forward paths.

In the following section, an algorithm is presented to obtain an expression of the form

$$X_m = \prod_{i=m}^{n-1} A_{i(i+1)} X_n$$

where feedback and feed forward paths are contained between the nodes n and m.

It is clearly seen that the value of a_o/b_o can be obtained easily by putting $s = 0$ in the expression for G(s). Therefore, for all practical computing purposes, value of the quantity a_o/b_o can be obtained by treating transmission functions as real numbers rather than functions of complex frequency and the variables associated with the nodes can be treated as functions of time. The multiplication property is then to be applied either directly or after simplification using the algorithm proposed in the following section.

2.3. TACKLING FEED-BACK AND FEED-FORWARD PATHS-A HIERARCHY BASED METHOD

Branches described by transmission functions of the form $A_{ij}|j \neq i+1$, are either feed-back or feed-forward branches. The signal flow graph may contain only feed-back branches or only feed-forward branches or both in addition to branches describable with transmission functions of the form $A_{ij}|j = i + 1$.. Thus there may be three cases which are as follows:-

case(i) Only feed-back branches are present

Let P be an ordered set of all feed-back branches. The ordering is done according to the following rule:-

#1

If $A_{i_1j_1}$ and $A_{i_2j_2}$ are any two members of P then $A_{i_2j_2}$ succeeds $A_{i_1j_1}$ if (a) $j_1 > j_2$ or (b) $i_1 < i_2 | j_1 = j_2$.

It is to be noted that condition (b) is merely a matter of convention. In such a case, the order in which the two entries are placed makes no difference when only feed-back branches are present. However, it is clearly seen from condition (b) that two feedback branches $A_{i_1j_1}$ and $A_{i_2j_2}$ having $j_1 = j_2$ affect the same series branch viz. $A_{(j-1)j}$ where $j_1 = j_2 = j$.

If more than one $A_{ij} \in P$ affect the same branch A, then the value of the transmission function A after the m^{th} branch is processed is given by

$$(A_{(j-1)j})_m = \frac{1}{\frac{1}{(A_{(j-1)j})_{m-1}} - \frac{\Delta_{ij}}{(A_{(j-1)j})_o}}$$

where $(A_{(j-1)j})_o$ is the original value of the transmission function before linearization and Δ_{ij} is defined by the following relation:-

$$\Delta_{ij} = \left(\prod_{k=j}^{i-1} A_{k(k+1)}\right) \cdot A_{ij}$$

In calculating the value of Δ_{ij} the most recent values of all the transmission functions involved are used.

case(ii) Only feed-forward branches are present.

Let Q be an ordered set of all feed-forward branches. The ordering is done according to the following hierarchy:-

If $A_{i_1 j_1}$ and $A_{i_2 j_2}$ are two successive members of Q, then $A_{i_2 j_2}$ succeeds $A_{i_1 j_1}$ if (a) $j_1 < j_2$ or (b) $i_1 < i_2 | j_1 = j_2$.

Again, the condition (b) is a matter of convention and need not be followed strictly when only feed forward branches are present in addition to the series branches.

Again, if more than one $A_{ij} \in Q$ modify a single series branch $A_{(j-i)}$ then the modified value of the transmission function of that series branch after the m^{th} such feed-forward branch is processed is defined by the following relation:-

$$(A_{(j-i)j})_m = (A_{(j-i)j})_{m-1} + (A_{ij})/\Delta_{ij}$$

where $\Delta_{ij} = (\prod_{k=i}^{j-2} A_{k(k+1)})$.

For m = 0 the value of $A_{(j-i)j}$ is the original value of the series branch transmittance before linearization. In calculating the value of Δ_{ij} the most recent values of transmittances of the branches involved are used.

case(iii) When both feed-back and feed-forward branches are present.

In this case the processing algorithm is slightly complex. A few definitions are proposed for tackling such operations.

(a)Containment of branches. A feed-back branch $A_{i_1 j_1}$ is said to be contained by another feed-forward branch $A_{i_2 j_2}$ if $i_1 < j_2$ and $j_1 > i_2$.

(b)Operation FB. If P be an ordered set of feed-back branches and if Q be an ordered set of feed-forward branches, then operation FB consists of application of operations formulated in case(i) using values modified by members of P only in the calculation each Δ_{ij}.

(c)Operation FF. If Q be an ordered set of feed-forward branches, then operation FF consists of application of operations formulated in case(ii) to all members of Q. However, in calculating the value of Δ_{ij} for a branch $A_{ij} \in Q$, the following steps are involved:-

(i)Demarcation of another ordered set $P' \subseteq P$, each member of P' being contained by the feed-forward branch in question.

(ii)Application of operation FB to the set P' to obtain modified values. In calculating the value of Δ_{ij} the modified values obtained through this step and those obtained in course of operation FF only are to be used.

Finally, if m number of elements in P and m' number of elements in Q modify a particular series branch $A_{(j-i)j}$, then the final modified value of the branch transmittance $A_{(j-i)j}$ will be

$(A_{(j-i)j})_{final} = (A_{(j-i)j})_{m'} . C_m$ where $(A_{(j-i)j})_{m'}$ is obtained through operation FF and $C_m = (A_{(j-i)j})_m / (A_{(j-i)j})_o$ is obtained through operation FB.

EXAMPLE

Let us consider the signal flow graph shown in fig.2.1

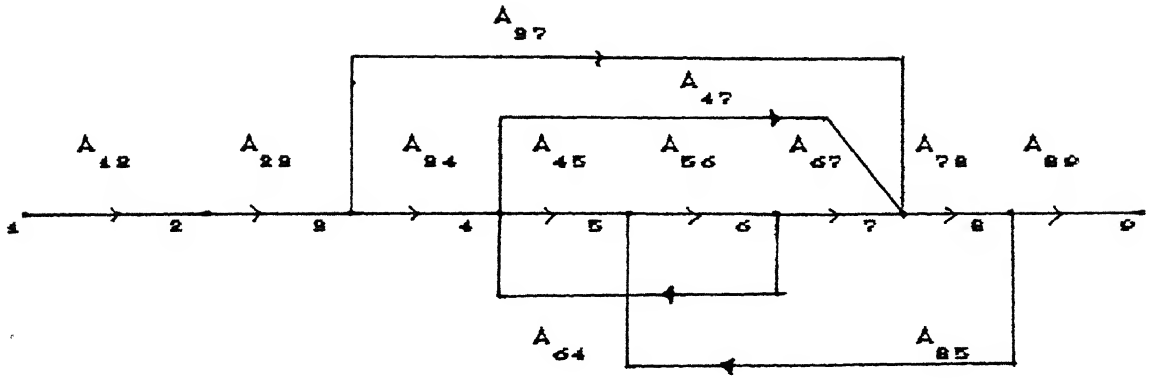


Fig.2.1. A signal flow graph showing feed-forward and feed-back branches.

For the above graph,the feed-back branches are A_{85} and A_{64} and applying rule #1,the order in which they are to be processed is:- (i)A_{85} and (ii)A_{64}.

For the above graph,the feed-forward branches are A_{27} and A_{47} and applying rule #2,the order in which they are to be processed is:- (i)A_{27} and (ii)A_{47}.

Operation FB gives:-

$$(A_{45})_1 = (A_{45})/(1 - \Delta_1) \text{ where } \Delta_1 = A_{56} \cdot A_{67} \cdot A_{78} \cdot A_{85}$$

$$(A_{34})_1 = (A_{34})(1 - \Delta_1)/(1 - \Delta_1 - \Delta_2) \text{ where } \Delta_2 = A_{45} \cdot A_{56} \cdot A_{67}$$

Operation FF gives:-

First feed-forward branch is A_{37}

It contains the feedback path A_{64}

Applying operation FB on $[A_{64}]$, we have,

$$(A_{67})_1 = A_{67} + A_{37}(1 - \Delta_2)/(A_{34} \cdot A_{45} \cdot A_{56})$$

The second feed-forward branch is A_{47} and it also modifies A_{67}

So, we have,

$$(A_{67})_2 = A_{67} + A_{37}(1 - \Delta_2)/A_{34} \cdot A_{45} \cdot A_{56} + A_{47}/A_{45} \cdot A_{56}$$

Using modified values

$$X_9 = \{A_{12} \cdot A_{23} \cdot A_{34} \cdot A_{45} \cdot A_{56} \cdot A_{67} \cdot A_{78} \cdot A_{89} / (1 - \Delta_1 - \Delta_2)$$
$$+ A_{12} \cdot A_{23} \cdot A_{37} \cdot A_{78} \cdot A_{89} \cdot (1 - \Delta_2) / (1 - \Delta_1 - \Delta_2)$$
$$+ A_{12} \cdot A_{23} \cdot A_{34} \cdot A_{47} \cdot A_{78} \cdot A_{89} / (1 - \Delta_1 - \Delta_2)\} \cdot X_1$$

which is precisely the result which Mason's gain formula for signal flow graph gives.

The above formulation has been experimented on a large number of graphs. However, it is quite possible that some configuration exists for which it fails. One disadvantage of this is the inability to tackle self loops. One great advantage of it is the ability to compute intermediate nodal values in cases where feed-back and/or feed-forward branches are also present.

values,has been used advantageously for fault simulation. Since the system configuration is fixed,a fault corresponds to a change in the transmittance of one or more branches. The change in intermediate values and the terminal value serves as an indication of a fault when they deviate from the normal.

2.4. SOME COMPONENTS AND THEIR REPRESENTATION AS SUB-GRAPHS

Some of the common mechanical and electrical components mostly required for electromechanical systems have been modeled as sub graphs. These sub graphs when connected as required build up the signal flow graph for the entire system. However,the menu of such components has been purposefully kept confined to a few selected components only as this simulator was primarily developed to simulate the electro mechanical shut-down rod system for the CANDU reactors which required a few selected components. The respective components and the sub graphs corresponding to each is as follows:-

(i) **Motor.** The motor modeled here is a dc servo motor in the armature controlled mode with constant field excitation. The variable associated with the first node is the input voltage v_i. The second node represents the voltage across the armature after considering the feed-back due to the back e.m.f of the motor. The third node represents the armature current. The fourth node represents the motor torque and the fourth represents the motor speed. The branch transmittances are as follows(neglecting the 's' terms):-

$A_{12} = 1.0$

$A_{23} = 1/r_a$ where r_a is the armature resistance

$A_{34} = K_m$ where K_m is the motor torque constant

$A_{45} = 1/F$ where F is the friction factor.

$A_{56} = 1.0$

$A_{52} = -K_m$ (this is also the e.m.f constant)

Note:- The value of the effective friction factor referred to the motor shaft can be obtained at steady state knowing the operating conditions of the motor.

(ii)Rotating parts like gears and pulleys. Each rotating part like an assembly of two gears or an assembly of a driver and a driven pulley is modeled as a three node sub graph. All the four nodes represent speed. The branch transmittances are as follows:-

$A_{12} = g$ where g is the ratio of driven shaft speed and driving shaft speed.

$A_{23} = 1.0$

For simulating faults in the gear/pulley assembly, the value of g is altered and for simulating faults associated with the driven shaft only, the value of A_{23} is altered. For pulleys it is further assumed that no slipping occurs.

(iii) Drum. A drum is modeled as a two node sub graph with the transmission function $A_{12} = 1.0$.

(iv) Controllers. All controllers are modeled as proportional devices. Each controller is represented by a two node sub graph. The first node and the second node both represent voltages. Voltage signal can be fed into the first node from any other node in the system signal flow graph. The branch transmittances $A_{12} = K_p$, where K_p is the controller gain.

(v)Potentiometers. A potentiometer is modeled as a two node sub graph with the transmittance $A_{12} = K$, where K is the potentiometer ratio.

(vi)NC and NO contactors. A normally closed NC contactor is represented by a two-node sub graph with A_{12} = 1 and a normally open contactor is represented by a two-node sub graph having A_{12} = 0 initially. The change in the value of the transmittances for contactors are brought about by the relay which acts on that contactor. Relays are modeled in the same ways as faults. A relay or a fault keeps record of two things viz. the node in the graph which they affect and the time instant(w.r.t the simulator clock) at which they occur. The simulator also permits simulation of processes involving timed operation of relays.

(vii)Triggers. A typical example of this is a comparator or a magnetic clutch. This is also modeled as a two node sub graph. The transmittance A_{12} swings from one predetermined value to another predetermined value as soon as the value of the variable associated with the first node crosses a certain predetermined threshold.

Feedback and feed-forward paths can only be modelled as scalers or potentiometers.

It is to be noted that the numbering of nodes for each sub graph mentioned above is being done w.r.t to the sub graph only. Fig. 2.1 shows the sub graph associated with a motor as an example.

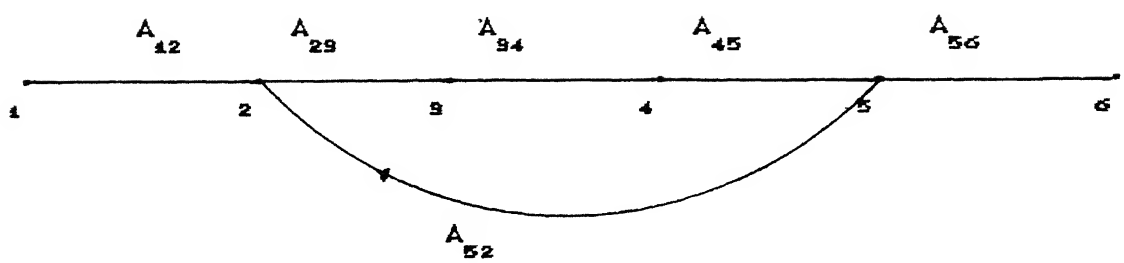


Fig. 2.1. Sub graph corresponding to an armature controlled d.c servo motor

2.5. SIMULATION OF EMSR SYSTEM FOR CANDU REACTORS.

The reactor shutdown mechanism for CANDU reactors consists of two shutdown systems viz. the EMSR(Electro mechanical shutdown rod) and the LPR(Liquid poison shutdown rod) mechanisms. After an abnormality worthy of reactor shutdown is detected by the monitoring instruments, the shutdown rod is expected to insert in the core completely in 2 seconds including the time required to communicate failure through instrument channels.

Fig. 4.1 shows the schematic representation of a single EMSR. From reactor physics considerations 12 such rods are required to introduce sufficient poison into the core in order to stop the fission chain reaction. There are 14 EMSRs triggered by 7 pairs of scram signal circuits.

The operation of a single EMSR is as follows. On receiving the scram signal, the rotating magnetic clutch MCR under the force of the release spring SR is disengaged. Under the weight of the shutdown rod, the cable on the drum unwinds freely and is guided by the pulleys P_1 and P_2 and the guide tube. The fall of the rod is impeded by the friction between the gears, between shafts and bearings and viscous drag due to moderator. To overcome the effect of this impeding factors, acceleration spring SPR1 is provided. Under normal condition the rod is pulled against SPR1 so that on release a very high initial velocity is achieved. At the end of the journey, the rod comes to rest on a support SP and to avoid vibrations a second spring SPR2 is provided.

The EMSR system is said to operate successfully, if 12 out of 1 rods reach the end support SP in guide tube within 2 secs. Becaus

rods reach the end support SP in guide tube within 2 secs. Because the rods are triggered in pair by 7 scram signal circuits, failure of two or more instrument channel out of 7 or failure of 3 out of 14 rods to insert completely in the core is considered as EMSR failure.

However, there is one very vital point which is to be accounted for. The spring is an active device. So far in the simulation algorithm it is assumed that the driving function is always applied at the input node. In this case, as soon as the magnetic clutch releases, the input terminal of the spring replaces input node and acts as the pseudo input node. A ramp approximation is made for the motion of the rod through the moderator. However, the actual trajectory of the moderator is something quite different.

For simulation purposes, the signal flow graph of the system apart from the spring loaded shutdown rod is constructed using the sub graphs for each and the spring-loaded shutdown rod is treated as a separate entity as a whole. The input terminal of the spring, is connected to some rotating member, in this case the pulley. It is to be noted that the angular displacement of the rotating member depends on the vertical displacement of the rod and so the magnitude of the pseudo input node driving function can be computed.

Let $X_i(t)$ be the nodal value at the spring output and let $X_i(t) = K.t$ rather than $K\delta(t)$.

Let $X_j(t)$ be the displacement of the rod from the initial position. For the present approximation $X_i(t) = X_j(t)$.

Let $X_k(t) = RX_j(t)$ be the reactivity produced in milli K by the displacement $X(t)$ at any instant t. R is assumed to be constant.

It is to be noted that the $X_k(t)$ is negative for a positive $X_j(t)$. $X_j(t)$ is positive for motion of the shutdown rod into the core.

If 'L' be the displacement of the rod at $t = t_f$, where t_f is the time of fall required and ρ_f be the value of $X_k(t)$ at $t = t_f$, we can define $\delta X_j(t) = L/t_f$ and $\delta X_k(t) = \rho_f/t_f$.

Approximating $X_j(t_i) = X_j(t_{i-1}) + \delta X_j(t)$ where $X_j(t_n) = L$, we have $X_k(t_i) = X_k(t_{i-1}) + \delta X_k(t)$, and $X_k(t_n) = \rho_f$, n beimg the number of time steps considered and $X_j(t_o) = X_k(t_o) = 0$.

Considering a single group of delayed neutron precursor only, the neutron population at any time t following a step change in reactivity ρ, from steady state, is given by the equation $n(t) = n_o[\ A.\exp(\alpha_1 t) - B.\exp(\alpha_2 t)]$ where A, B, α_1 and α_2 are defined in the following way:-

$$A = \beta/(\beta - \rho);$$

$$B = \rho/(\beta - \rho);$$

$$\alpha_1 = \lambda\rho/(\beta - \rho);$$

$$\alpha_2 = (\rho - \beta)/\Lambda;$$

and

$\beta \rightarrow$ fraction of delayed neutrons produced during fission compared to the total(delayed + prompt). Typically, $\beta = 0.0065$.

$\rho \rightarrow$ magnitude of the step change in reactivity.

$\lambda \rightarrow$ mean ecay rate of the delayed neutron precurors, typically $\lambda = 0.08\ sec^{-1}$.

$\Lambda \rightarrow$ neutron production time, typically 0.1 ms.

$n_o \rightarrow$ neutron population at steady state.

The reactivity change during the shutdown process can be

considered as a step change if the time taken by the rod to reach the end support is very small. Since the time is appreciable(approximately 2 sec.) this step approximation is not valid. The reactivity, however remains constant after the rod reaches the end support and rests there. The actual variation of the reactivity during shutdown is governed by the motion of the rod itself. For the sake of our convenience, the reactivity change has been approximated as a ramp function during the journey of the rod through the moderator. To calculate the neutron populations at different time instants during the journey of the rod through the moderator, the folowing formula is applied: -

$$n(t_i) = n(t_{i-1}).[A'.\exp(\alpha_1' t) - B'.\exp(\alpha_2' t)]$$

where $A' = \beta/(\beta - X_k(t_i))$;

$$B' = \dot{X}_k(t_i)/(\beta - X_k(t_i))$$

$$\alpha_1' = \lambda.X_k(t_i)/(\beta - X_k(t_i))$$

$$\alpha_2' = (X_k(t_i) - \beta)/\Lambda$$

The above formulation is not strictly valid since, the momenta step change in reactivity is introduced into a reactor at steady state, a prompt jump in neutron population occurs. But in the present case, the reactivity changr is on the negative side. So for small time step and small value of time step this approximation gives fairly satisfactory results. Moreover, since both the normal and the simulated responses are calculated under the same approximations, there is sufficient qualitative difference between the response in the normal state and that in the faulty state to enable detection of faults. The values of ρ_f, 'l' and 't_f' are to be provided by the user when the nuclear interface is built.

CHAPTER THREE

THE DIAGNOSIS ALGORITHM

Diagnosis is based on a search of the system fault-tree which is represented by an AND-OR tree and is entered by the user in the form of a set of Boolean equations each representing the causal relationship between a fault and its causes. The entries are in the Sum of Terms form where the left hand side represents some intermediate event(symptom) and the right hand side is an expression involving disjunction of a number of terms, each term being an event or a conjunction of two or more events. The traversal algorithm is a three tier process starting with the TOP event as the root and gives as its output the set of faults which have occurred. The first step is based on purely deductive reasoning. The second step involves MYCIN like uncertainty handling techniques to demarcate a set of most probable faults from the set of all possible faults about which some ambiguity exist. The third step is based on user interrogations and is used to identify the faults which have definitely occurred.

3.1 KNOWLEDGE REPRESENTATION

The knowledge is represented in the form of an AND-OR tree. The TOP event is the root of the tree. Each node in the tree has at least two attributes associated with it depending on whether it represents an INTERMEDIATE EVENT or a BASIC

EVENT. These are:-

(a)INSTRUMENTATION STATUS if it is an intermediate event. It denotes whether or not there is an instrumented fault indicator for the intermediate event in question. If there is an instrumented fault indicator, then it is possible to know definitely the state of the intermediate event in question, else it may or may not be possible to check for the state of the event in question.

(b)PROBABILITY OF OCCURRENCE if it is a basic event. This attribute is not static for a node and has to be computed.

(c)OPEN or CLOSED attribute to denote whether the node has to be checked(when it is still OPEN) or not(when it is CLOSED). A node is said to be **OPEN** till no query has been made about the status of the event represented by it.

Thus, for two nodes i and j representing two events S_i and S_j respectively and with a link L_{ij}, the direction being from node i to node j, the semantic significance is:-

if S_i occurs then there is a probability P_{ij} that S_j is the cause for it.

This is similar to MYCIN rules. But a set of static rules cannot be used in this case as the aforesaid probability P_{ij} is dynamic and has to be normalized over the set of possible faults having some ambiguity since the probability of occurrence of a fault depends on the equivalent failure rate of the components associated with it, the time of operation, and of course the conditions associated with it can vary widely. Hence normalization over a finite domain is necessary. Another

advantage of this representation, which is quite obvious from the first,is the ability to use the same graph repeatedly and computing the probability values as and when required rather than forming a fresh set of rules each time as required by a static set of representations like MYCIN.For example,even after a certain fault has occurred,a static set of rules will assign the same probability of occurrence to it for the same set of symptoms,though in real life the probability of occurrence of the fault in the near future is much lower.In such a case,in order to overcome this problem,a fresh set of probabilities are to be assigned to each fault and a new set of rules is to be formulated.

However,for implementation is done by predicates to store information about causal relationships between the faults and the symptoms.

Let T → TOP event;

Let $S \equiv [s_1, s_2, s_3, \ldots\ldots\ldots, s_n]$ denote the set of all intermediate events.

Let $E \equiv [e_1, e_2, e_3, \ldots\ldots\ldots\ldots, e_n]$ denote the set of all possible basic events.

In the light of the above discussions,the following predicates are defined:-

(i) can_occ_frst(s_i,sE_i) where s_i is any intermediate event and $sE_i \in p(E)$ where p(E) is the power set of E.

(ii)possible_frst_if(E_i,sS_i) where E_i is any basic event and $sS_i \in p(S)$,p(S) being a power set of S.

The semantic significance of the above two predicates is

brought out by the following rules:-

(1)can_occ_frst(s_i,sE_i)$\wedge$($\neg$occurs(s_i)) $\supset$ ($\neg$occurs(e_j)|$\forall$ e_j in sE_i).

The rule states that if a certain intermediate event is not seen then none of the basic events causing it could have occurred.

(2)possible_frst_if(e_i,sS_i) $\wedge$ (occurs(e_i)) $\supset$ (occurs(s_j) |$\forall s_j$ in sS_i)

The second rule states that if any basic event occurs,then all the intermediate events caused by it occur.

As an example, let us consider the following formulations defining the cause effect relationship between a set of symptoms(intermediate events) s_1,s_2 and s_3 and a set of causes(basic events) e_1,e_2,e_3,e_4 and e_5.

Example 1.

$$s_1 = e_1 + e_2 + e_3$$

$$s_2 = e_3 + e_4$$

$$s_3 = e_1 + e_5$$

The corresponding predicates will be:-

can_occ_frst(s_1,[e_1,e_2,e_3])

can_occ_frst(s_2,[e_3,e_4])

can_occ_frst(s_3,[e_1,e_5])

possible_frst_if(e_1,[s_1,s_3])

possible_frst_if(e_2,[s_1])

possible_frst_if(e_3,[s_1,s_2])

possible_frst_if(e_4,[s_2])

possible_frst_if(e_5,[s_9])

The formulation of the above rules require that the each boolean relationship representing some cause-effect relationship is a function involving disjunction of a set of basic events only. This is the simplest case and it is the desired form of representation For handling cases involving disjunction and/or conjunction of basic and/or intermediate events some transformations are necessary.

The user, however, enters the fault-tree in the form of a set of boolean equations. Two cases have been dealt with separately. These are as follows:-

case(a). A set of intermediate events is the cause for another intermediate event s_i and the logical formulation is consistent.

A predicate causes(intermediate event, set of basic events, set of intermediate events) is defined to mean that the first argument is caused by any basic event in the second or by any intermediate event in the third.

The following transformation rules may be applied in this case to transform the formulation to a form which can be expressed with the help of the two predicates can_occ_frst and possible_frst_if:-

#1causes(s_i,sE_i,sS_i) $\supset$ ($\forall$ $s_j \in sS_i$ causes(s_i,sE_i,[s_j])).

#2causes(s_i,sE_i,[s_j])$\wedge$(can_occ_frst(s_j,sE_j)) $\supset$
(can_occ_frst(s_i,sE_k | $sE_k := sE_i \cup sE_j$).

The truth table in the simplest form can be obtained by

applying transformation rules #1 and #2 to elements of S whenever required.

case(b). This is a more general case where the boolean equation involving the causal relationship between the symptoms and the causes involves disjunction and/or conjunction of intermediate and/or basic events. A typical formulation in the clausal form may be considered.

$s_i :- s_{k_1}, s_{k_2}, s_{k_2}, \ldots\ldots\ldots, s_{k_m}$.

$s_i :- s_{l_1}, s_{l_2}, s_{l_3}, \ldots\ldots\ldots, s_{l_n}$

.

where s_k, s_{l_q} can be a basic event or an intermediate event.

In this case the following set of formulations are attempted to resolve the conjunctions:-

Transformation #1.

The aforesaid clausal formulation is to be rewritten as:-

$s_i :- s^*$.

$s_i :- s^*$.

$\forall$ p in [1,m] $s_{kp} :- s^*$.

$\forall$ q in [1,n] $s_{lq} :- s^*_l$.

Transformation #2.

Each of the events s_k, s_{l_q} are to be treated as intermediate events. They must have instrumented indicators to determine their state uniquely. The events s_k^*, s_l^* are to be treated as basic events of Zero equivalent failure rate to avoid traversal of closed nodes. Transformation rules #1 & #2 cited in case(a) are to be applied now and the truth table in the desired form is to be obtained and standard predicates defined may be used.

It is clearly seen that the conjunctions are very costly from instrumentation point of view.

The transformations #1 & #2 can be applied to any finite number conjunctions associated with s_i.

EXAMPLE 2.

$$s_1 = a.b + c$$

$$s_2 = d$$

$$s_3 = a + e$$

this can be represented by an alternative formulation which is not exactly logically equivalent but produces approximately the same trajectory.

$$s_1 = s^* + c$$

$$a = s^*$$

$$b = s^*$$

$$s_2 = d$$

$$s_3 = a + e.$$

The basic events originally represented by a and b should be treated as intermediate events, s^* should be treated as a basic

event and there should be instrumented fault indicators for both a and b.

The logical inequivalence lies in the fact that the above formulation indicates that s^* cannot occur if both a amd b do not occur, which is quite true, but fails to utilise the fact that if s^* occurs then both a and b must have occurred. However, since s^* is not checked at all, the second implication has no significance.

3.2 SEARCH BASED ON DEDUCTIVE REASONING

Let S_a be set of all intermediate events which have instrumented fault indicators;

Let $S_b := S - S_a$ be set of all intermediate events which do not have instrumented fault indicators;

It is to be noted that all events in S_b can be checked by the user when required and three answers 'yes','no' and 'don't know' are possible to such a query.

Let $S_d \in S_a$ be set of intermediate events which have occurred;

Therefore, $S_e := S_a - S_d$ is set of intermediate events which have not occurred.

The following routines are then used to separate out the set of faults which are definitely known to have occurred and the set of faults which are definitely known to have not occurred. The remaining subset of E is the set of faults about which some ambiguity exists.

Routine #1.

$\forall s_i$ in S_e can_occ_frst$(s_i, sE_i) \supset (E_i := E_{i-1} - sE_i)$ where E_i is the set of possible basic events at the time of processing the i^{th} intermediate event in S_e and it is initialized to E. This rule states that the causes corresponding to each intermediate event in S_e can be deleted from the set of possible faults. The predicates of the form can_occ_frst(intermediate event, set of basic events) are to be updated and sE_i is to be subtracted from all these using the following rule for all elements in S_d

$\forall s_j \in S_d$, can_occ_frst$(s_j, sE_j) \supset$ (can_occ_frst(s_j, sE'_j)

where $sE'_j := sE_j - E_a$

Routine #2.

$\forall s_i$ in $S_d \exists$ possible_frst_if$(e_k, [s_i]) \supset (E_{d_i} := E_{d_{i-1}} + [e_k])$ where $e_k \in E$, and E_{d_i} is the set of basic events which are definitely known to have occurred at the time of processing the i^{th} intermediate event in S_d and it is initialized to a null set.

This rule states, if a particular intermediate event occurs and if it can be caused by one particular basic event only, then that basic event definitely occurs.

The set $E_b := E - (E_a + E_d)$ be the set of faults about which some ambiguity exists E_a and E_d being the sets of faults which are definitely known to be absent and the set of faults which are definitely known to be present respectively and they are obtained through routine #1 and #2. Thus, for example 1, if $S_a \equiv [s_1, s_2]$ and $S_d \equiv [s_2]$, then $E_a \equiv [e_1, e_2, e_3]$, $E_d \equiv [e_4]$ and $E_b \equiv [e_5]$

3.3 FORMATION OF A SET OF MOST PROBABLE FAULTS

If the set E_b is not a null set,then further reduction of E_b may be possible using probabilistic techniques.The method followed in this case is a modification of MYCIN.

However,the major deviation from MYCIN lies in the fact that symptoms in this case provide only positive inferences about the occurrence of faults, unlike MYCIN in which the rules are framed in such a manner that . particular symptom can add to belief or disbelief about a particular hypothesis.Accordingly,it computes only the belief measure for a particular fault when the set of symptoms dictate that the fault is possible.Computing disbelief is not required because the method is designed to be used on a set of faults all of which are possible for a given set of symptoms.

$\forall e_i$ in E_b,

let $p(e_i) = 1.0/n$ denote the apparent probability of occurrence of a particular fault e_i considering equal probability for all faults,n being the number of elements in E_b;

let $p(e_i)_a = 1 - e^{-\lambda_i T_i}$ denote the actual probability of of occurrence of basic event e_i,λ_i being the failure rate associated with basic event e_i and T_i being the corresponding time of operation for the components associated with it;

let $p(e_i|T)$ denote the normalized probability of occurrence of the event e_i if the TOP event T occurs.

In the computation of $p(e_i)_a$, we assume the following:-

(1)A fault is a permanent state and can only be removed through repairs and/or replacements and not by itself.

(2)Components associated with a fault get a fresh life once a particular fault is rremoved either through repair or through component replacement.

The value of the normalized probability $p(e_i|T)$ for a basic event $e_i \in E_b$, can be computed using the following relation:-

$p(e_i|T) = p(e_i)_a / (\sum_{k=1}^{m} p(e_k)_a)$ where m is the number of elements in E_b.

The values of $p(e_i|T)$ and $p(e_i)_a$ thus obtained are used in the computation of a certainty factor to indicate whether it is worth searching for the particular basic event or not.

The certainty factor is defined in the following way:-

C.F = 1 if $p(e_i) = 1$;

otherwise C.F = $p(e_i|T) - p(e_i)$.

The C.F is computed for all e_i in E_b basic events with C.F< 0 are deleted from the set E_b. Thus $E_v := E_b - E_c$ represents the set of most possible faults, E being the subset of less probable faults. E_v defines the domain of search for the next tier.

3.4 ELIMINATION BY USER INTERROGATION.

Let S_x define the initial set of intermediate events about which no conclusion can be drawn. Elements may be added to it or

deleted from it in course of the interrogations.It is initialized to the null set.

Let E_x define the initial set of basic events about which no conclusion can be drawn.It is initialized to a null set.

Let E_o define the set of OPEN basic events and it is initialized to E_v.A basic event(or more specifically the node representing it) is said to be OPEN if it is not a member of E_a or E_d or E_x.

Let S_o define the set of OPEN intermediate events.An intermediate event is said to be OPEN if it is not a member of $S_d$or S_a or S_x.

$\forall$ e_i in E_v,the following sequence of operations is performed:-

(i) test for occurrence of e_i:-

$(e_i \text{ in } E_o) \wedge (\text{possible\_frst\_if}(e_i, sS_i)) \wedge (s_j \notin S_e \mid \forall\ s_j \text{ in } sS_i)$.

While testing the individual $s_j \in sS_i$,the following are to be considered:-

(a)s_j is to be tested if $s_j \in S_o$.

(b)if occurs(s_j) then $S_d := S_d + [s_j]$.

(c)if not occurs(s_j) then $S_a := S_a + [s_j]$, rule #1 cited in section 3.2 is to be applied on s_j.

(d) if it cannot be concluded whether occurs(s_j) or not occurs(s_j) then $S_x := S_x + [s_j]$.

(ii) after testing e_i, any one of the following is to be done:-

(a) if occurs(e_i) and possible_frst_if(e_i,sS_i) $S_d := S_d + sS_i$, $S_x := S_x - sS_i$, $E_d := E_d + [e_i]$.

(b) if not occurs(E_i) then $E_a := E_a + [e_i]$.

(c)if it cannot be determined whether occurs(E_i) or not occurs(E_i) then $E_x := E_x + [e_i]$.

If after testing all the elements of E_v, E_d = [] then less probable faults are to be considered.For this purpose,E_v is reinitialized to E_c and the sequence of operations formulated in (i) and (ii) are to be performed again.

3.5 MERITS AND DEMERITS OF THE STRATEGY.

It is evident from the above discussions that the strategy gives best results in cases where the fault-tree can be represented by an OR graph only.AND terms are difficult to tackle in this case from instrumentation point of view.Moreover,if any boolean equation describing the causal relationship between a symptom and its causes contains AND term on its RHS,then instrumentation is required for all the variables(either basic or intermediate events) that are involved in that term.Furthermore,the strategy fails to use the knowledge represented by unary negation.This is because of the fact that unary negation cannot be represented by AND or OR functions.In fact,representation of unary negation in an AND-OR graph requires an additional operator. However,for MYCIN like diagnosis problems,the fault trees encountered can be represented by a graph having OR links only,each link being directed from the effect to the cause.Negations and conjunctions are rarely encountered except for systems like purely digital systems.Even for such systems,MYCIN like set of

rules can describe the cause-effect relationship between symptoms and the causes quite effectively but representing negation requires concept of 'dis-belief' also. Typically,a combination of symptom(s) point to a set of faults each with a certain probability.This method is therefore usable for systems whose fault-tree can be represented by simple MYCIN like rules.

The formulation suggested in section 3.1 for dealing with the AND entries,though not exactly logically equivalent,does not give rise to logical inconsistency and does not search for the occurrence of a CLOSED node.For dealing with highly formal systems,e.g diagnosis of 'stuck at 0' or 'stuck at 1' faults,probabilistic techniques are not required and faults can be detected using deductive reasoning only.

CHAPTER FOUR

CASE STUDY ONE - EMSR SYSTEM OF CANDU REACTORS

The reactor shut down system for CANDU reactors consists of two systems viz. the Electromechanical Shutdown Rod(EMSR) and the Liquid Poison Rod(LPR) shutdown system. In this chapter, the fault-tree of the CANDU EMSR system is presented and the expected performance of the diagnosis algorithm for diagnosing faults in the aforesaid system has been put forward.

4.1 FAULT TREE FOR EMSR SYSTEM.

The EMSR system for CANDU reactors has been discussed in Chapter two. Fig. 4.1 is the schematic representation of a single electro mechanical shutdown rod. system. The EMSR system is said to operate successfully if twelve out of fourteen rods reach the end support SP in the guide tube within two seconds and rest on it. Because rods are triggered in pairs by seven scram signal circuits, failure of two or more instrument channels of the seven or failure of three out of fourteen rods to insert completely in the core without damaging the support is considered as EMSR system failure.

Therefore, failure of EMSR System ≡ (Failure of two or more of the instrument channels) OR (Failure in safe insertion of three or more out of fourteen rods).

The TOP event for the fault-tree of EMSR is defined as:-

FAILURE OF EMSR TO FALL TO THE BOTTOM IN TWO SECONDS AND REST ON THE END SUPPORT SP.

The fault-tree for a single EMSR is shown in fig. 4.1For diagnosis purposes using probabilistic techniques, it is necessary to have the failure probabilities associated with the basic

events. For the present case, the failure rate data have been taken from an earlier analysis[] which depends largely on WASH-1400. Failure rate corresponding to each basic event is tabulated in table 4.1. The AND - OR graph corresponding to the fault-tree is shown in fig. 4.2.

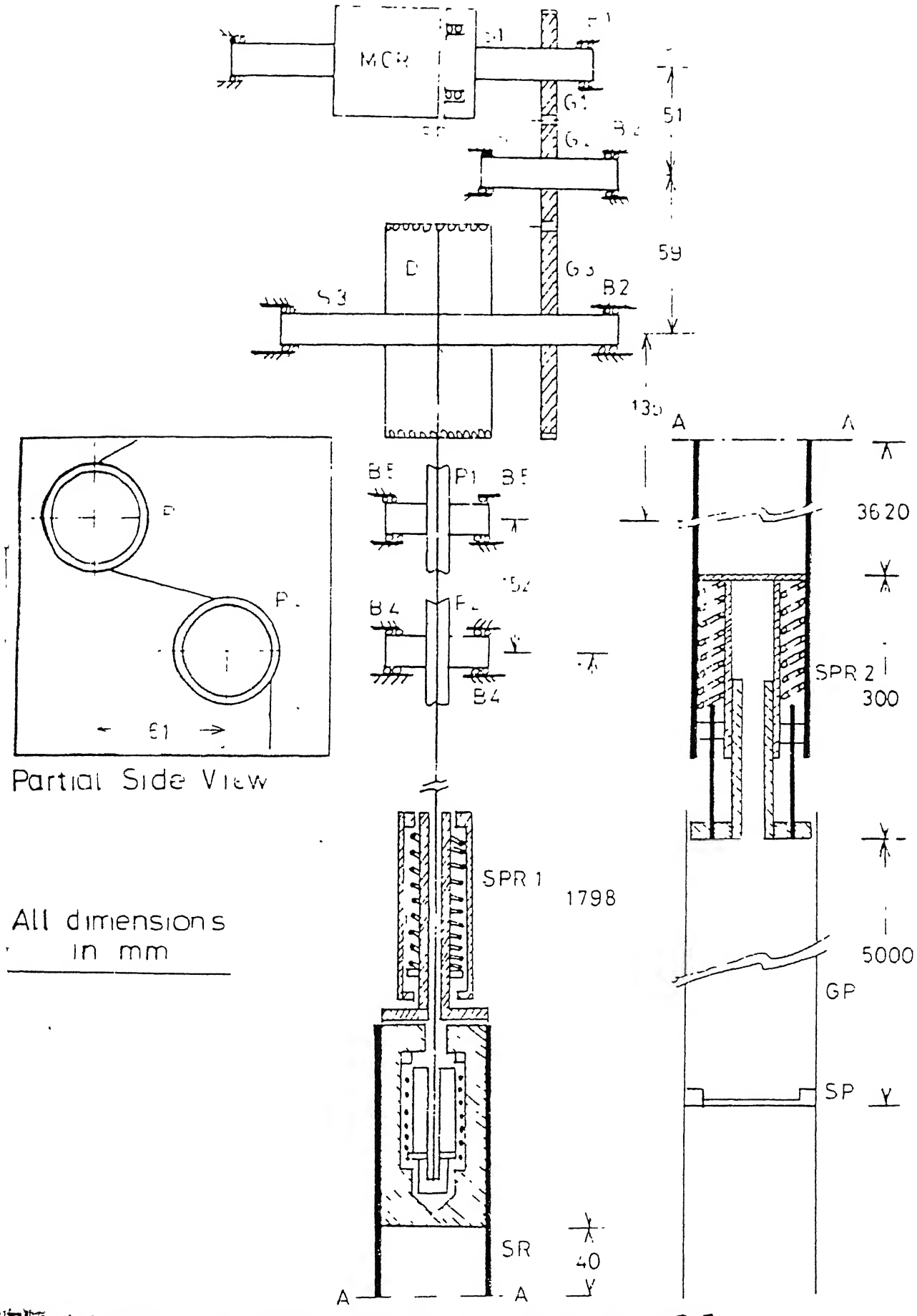


Fig. 4.1. Schematic representation of a single EMSR.[5]

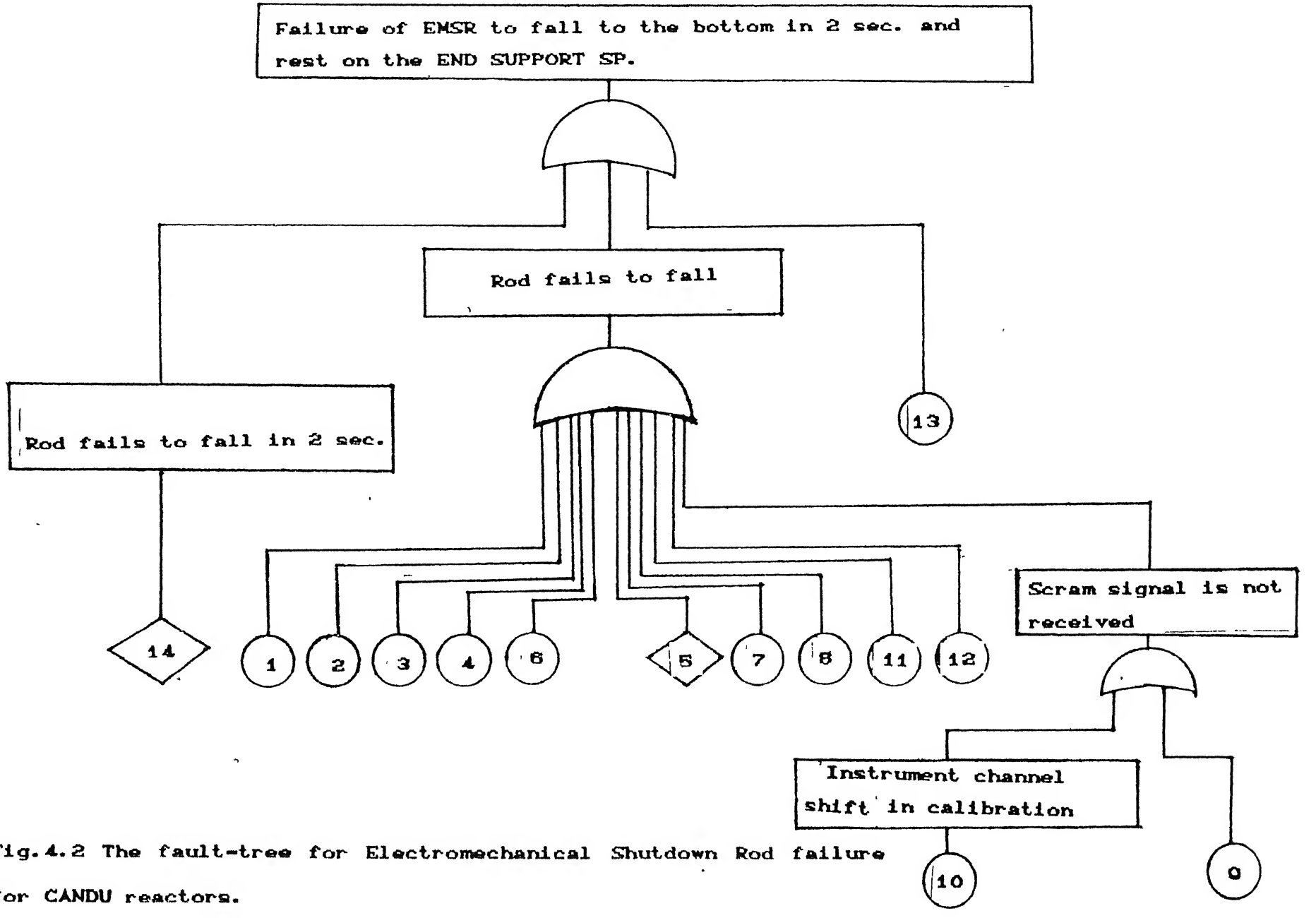


Fig.4.2 The fault-tree for Electromechanical Shutdown Rod failure for CANDU reactors.

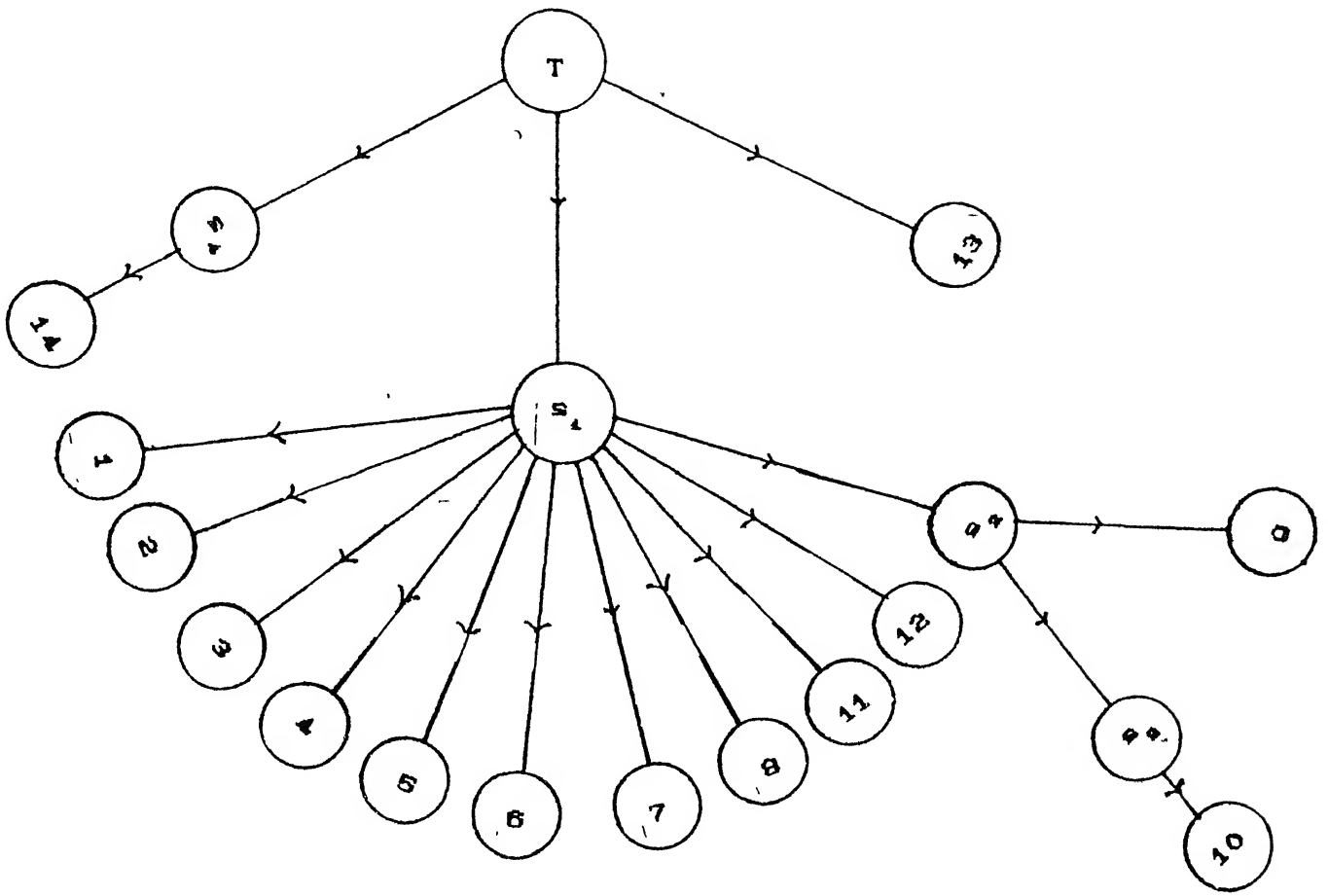


Fig 4.3 The AND-OR Tree constructed from the Fault-tree of fig.4.2

Note:- For description of Basic events, Table 4.1 is to be referred.

Table 4.1

Data on failure rate or unavailability of Basic Events of fault-tree of Electomechanical Shutdown Rod.

Event	Description	Failure rate
e1	shaft 2 fails to rotate	0 .609
e2	gears G2 & G3 locked	0.12
e3	gears G1 & G2 locked	0.12
e3	shaft 3 fails to rotate	0.609
e5	rope is stuck in a groove	not known
e6	shaft 1 fails to rotate	0.609
e7	shaft 4 fails to rotate	0.609
e8	shaft 5 fails to rotate	0.609
e9	instrument channel fails	1.612
e10	shift in instrument channel calibration	48.33
e11	spring SR(a pair) fails to release	0.42
e12	accelerator spring fails to release	0.0241
e13	support SP breaks	0.024
e14	rod fails to reach support in 2 sec.	not known
e15	testing & maint. of instrument channels	1.2×10^{-5}
e16	testing of mech.systems	0.001
e17	maint.of mech.systems	2.138×10^{-9}

note:-

(1).Failure rate is expressed as a mean per 10^6hr.

(2).No failure rate data is available for events e_5 and e_{14}.

4.2 ANNOTATIONS PERTAINING TO AND-OR TREE FOR THE CANDU EMSR.

T → Top Event;

s_1, s_2, s_3 and s_4 are the intermediate events.

Following is the semantic significance of the symbols associated with the intermediate events:-

s_1 → Rod fails to fall;

s_2 → Scram signal is not received

s_3 → Instrument channel shift in operation.

s_4 → Rod fails to fall in 2 sec.

All of them are assumed to have instrumented fault indicators. The basic events e_{16} and e_{17} have not been included in the AND-OR tree.

There is no failure rate data corresponding to the basic events e_5 and e_{14}. To ensure detection of e_5 and e_{14}, $\lambda_5 = \lambda_{14} = \lambda_{max}$ where λ_{max} is the failure rate corresponding to the event with the highest failure rate in the set of all possible events that can be considered for probabilistic search. Owing to the extremely high failure rate value associated with e_{10} which is several orders of magnitude larger compared to others,it has been instrumented and has been linked to an intermediate event viz.s_3.It is thus evident that if s_3 is tested,then e_{10} will not be tested.e_{10} will therefore,never be considered for probabilistic search.Accordingly,$\lambda_{max} = 1.816/10^6$ hr which is equal to λ_6 It is to be noted that e_{14} will not be considered for probabilistic

search since its state can be determined by examining the state of S_4

.3. PERFORMANCE ANALYSIS OF THE DIAGNOSIS ALGORITHM FOR THE PRESENT CASE.

From the AND-OR tree it is evident that only discrete combinations of intermediate events is possible. Corresponding to each of these combinations, two indices M and M' are defined. M represents the number of basic events to be considered in the worst case when even low probability faults are to be considered. M' represents the number of basic events comprising the set of most probable faults. This is equal to the number of searches when at least one fault in the set of all possible faults is found to have occurred. It is to be noted that M represents the number of elements in the set of possible faults after the first tier of search and M' represents the number of elements in the set of most probable faults.

The indices M and M' are based on the following assumptions: -

(i) Time of operation associated with all faults is the same.

(ii) $\lambda_i T_i$ is a small quantity, where λ_i is the equivalent failure rate associated with the i^{th} fault and T_i is the time of operation associated with the i^{th} fault.

The valid combinations of occurrence of the intermediate events and corresponding values of M and M' are provided in Table 4.2. The formulation put forward in Table 4.2 is valid for the first occurrence of the TOP event. Thereafter, the number of searches required will be dictated by the faults themselves. But, if the

probabilistic nature of the faults is assumed to hold then the average worst case will not be much different from this.

TABLE 4.2

Table of valid combinations of intermediate events for the fault-tree of CANDU EMSR and corresponding values of performance indices.

SYMPTOMS				M	M'
S_1	S_2	S_3	S_4		
0	0	0	0	0	0
0	0	0	1	1	1
1	0	0	0	11	6
1	0	0	1	11	6
1	1	0	0	11	6
1	1	0	1	11	6
1	1	1	0	11	6
1	1	1	1	12	7

CHAPTER FIVE

CASE STUDY TWO- TROUBLE SHOOTING IN IBM PC

The diagnosis strategy developed for fault diagnosis has been applied for guided trouble-shooting in an IBM PC and can guide tests to track component level faults. The knowledge base covers double drive IBM PCs though some aspects of the IBM PC XTs are also covered by it. Fault trees were constructed for problems related to

(i) Starting up

(ii) Reading/Writing

(iii) Keyboard problems

However, the fault trees in none of the cases incorporate human errors and errors with coded diagnostic messages which are provided by the built in diagnosis software of the PC itself. Another important aspect of these fault trees is the fact that data is not available for a large number of basic events, but almost all of them can be tested to determine their state. The modularisation of the PC faults into separate modules makes trouble-shooting easier by selecting the appropriate problem areae.g Starting up, Reading/Writing, and Keyboard problems) but at the cost of some interrelated information.

5.1 DATA TREATMENT FOR TROUBLE SHOOTING THE IBM PC.

A database of failure rates of major electrical and electronic components has been prepared. Failure rate associated with the fault is then computed from the failure rates of the associated components. The data is obtained from [2].

For any system, if any one family of parts has a failure rate of λ_j then the system failure rate will be

$$\lambda_s = \sum_{j=1}^{n_f} n_j \lambda_j$$

In the above expression n_f is the total number of such families and n_j is the total number of components of a family which jeopardize the system. It is evident that the value of n_j depends on system configuration. More accurate the value of n_j, more expertly the system will diagnose the faults.

Setting $n_j = 1 \ \forall j$, we get the failure rate corresponding to a system comprising of one component of each kind. This has an advantage as well as a disadvantage. The advantage lies in the fact that the system description is very simple and it assumes no knowledge about the system configuration. But at the same time much of its power to tell apart more probable faults from the less probable ones is lost. Thus a diagnosis based on this simple model will fail to distinguish a fault associated with a single resistor

and a fault involving a collection of resistors on the basis of probabilities only unless the number of components is specified. The interface software designed to build the diagnosis knowledge base prompts the user for the number of components and assumes it to be unity by default.

For the present case, diagnosis is done mostly by setting n_j equal to unity as most of the faults are associated with individual components at particular locations only. Moreover, during the normal operation of a PC the RAM ics are used much more. Therefore, failure rates associated with RAM ics is multiplied by a factor(10 in the present case) to qualitatively tell apart failure of a RAM ic as a more probable fault than others.

Although, fault-trees have been constructed for all the four problem areas mentioned at the beginning of this chapter, analysis is done for start up related problems only. The sets of Boolean equations describing the cause-effect relationships for other problems are also presented. The analysis technique is similar to that for start up related problems and therefore is excluded for the sake of brevity.

For building up the cause effect relationship, some diagnosis aids avilable were also considered.

5.2 FAULT TREE FOR START UP RELATED PROBLEMS IN THE IBM PC AND ITS TREATMENT

The fault tree for start up related problems in the IBM PC is shown in fig.5.1 and the corresponding AND-OR search tree is presented in fig.5.2. Table 5.1 presents the failure probabilities associated with each basic event. The failure rates represent mean failure rate/10^9hrs.

5.3 ANNOTATIONS PERTAINING TO THE AND-OR SEARCH TREE

T → Top Event ≡ Failure to turn on and boot up when power is switched on.

s_1, s_2, s_3 are the intermediate events.

s_1 → Power light off and no response.

s_2 → Power light is on, but nothing on the screen.

s_3 → Power light is on, but drive will not boot a disk.

Failure rate data is not available for e_4, e_5, e_7 and e_8. Therefore $\lambda_4, \lambda_5, \lambda_7$ and λ_8 are each assigned an equivalent failure rate equal to λ_1 following the same line of logic presented in Chapter four.

All other assumptions remain the same as those formulated in Chapter four.

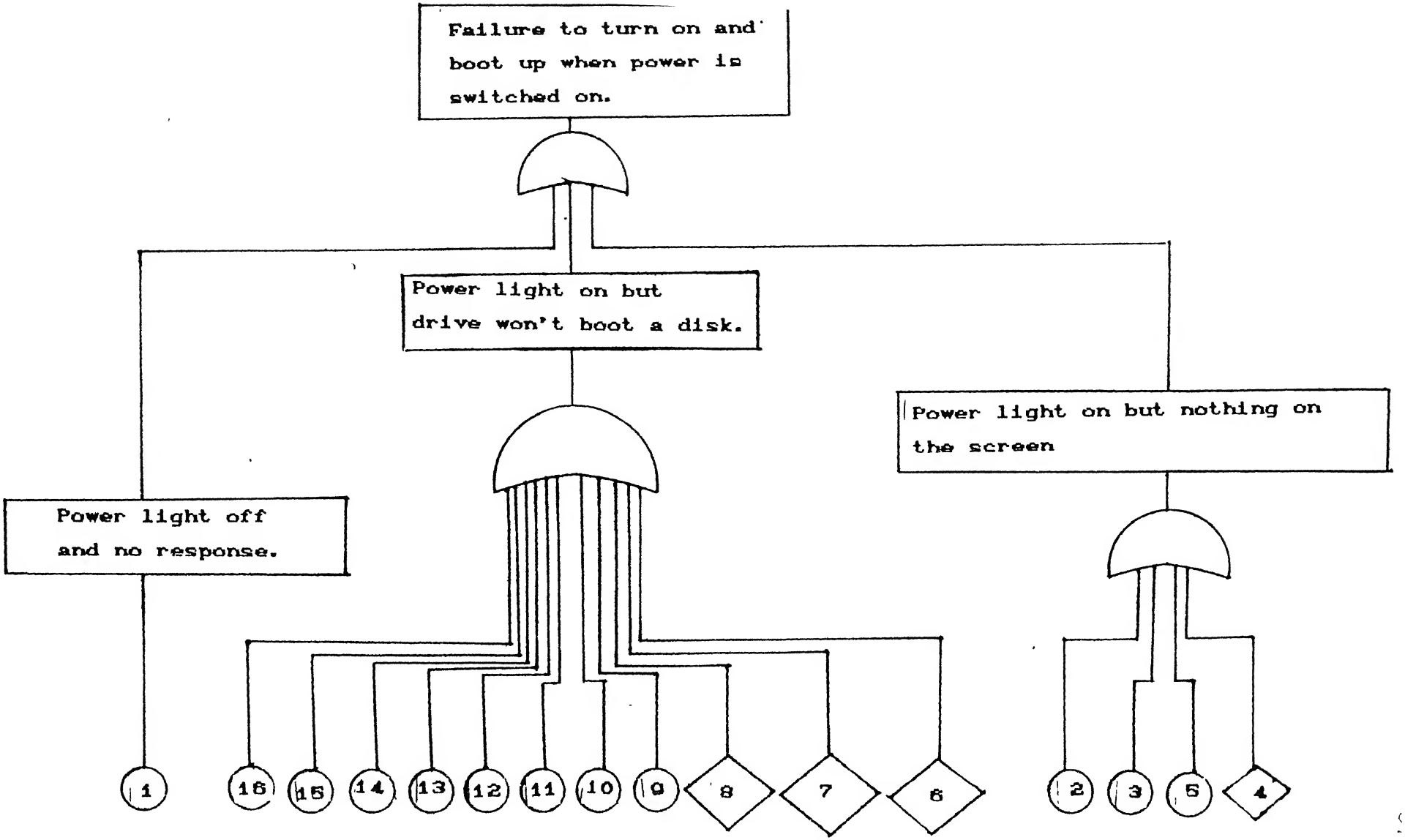


Fig.5.1. The fault-tree for start up related problems in IBM P.C.

Note:- Table 5.1 is to be referred for basic event descriptions.

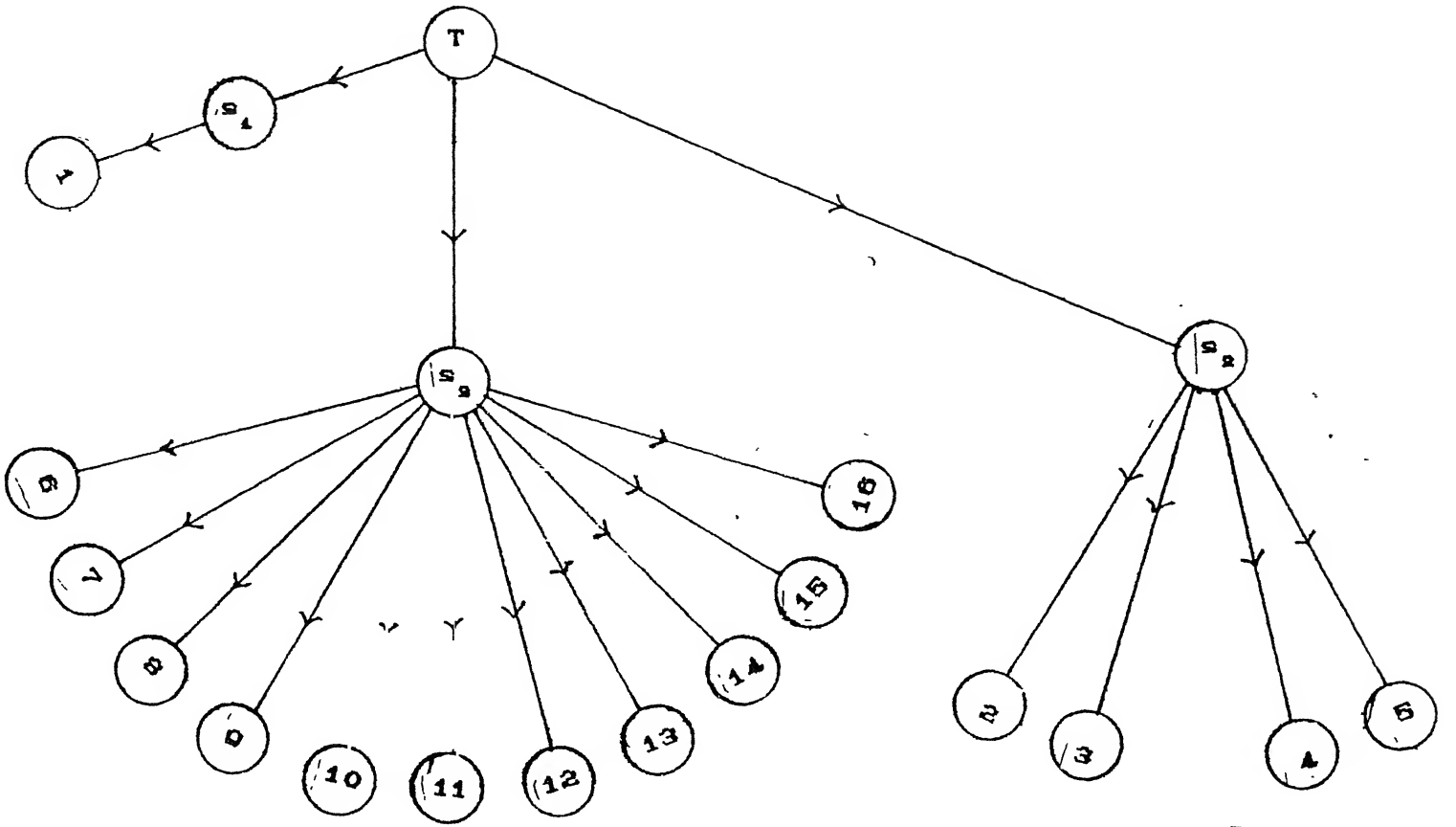


Fig.5.2 The AND-OR tree corresponding to fault tree in fig.5.1

TABLE 5.1

Data on equivalent failure rate associated with each Basic Event in the fault-tree for start up related problems in IBM PC.

Event	Description	Failure rate
e1	Power supply failure	0.1564
e2	Bad 8284	0.0703
e3	Bad 8088	0.0703
e4	Bad X-tal	not known
e5	Bad Rom	0.0703
e6	Bad disk	not known
e7	Bad data or disk dislodged	not known
e8	cable bad or loose	not known
e9	Bad 74s38 at 1f	0.0025
e10	Bad resistor at 5e	0.0025
e11	Bad 74ls86 at 5d	0.0703
e12	Bad 74ls74 at 5c	0.0703
e13	Bad 311 at 5b	0.0025
e14	Bad 592 at 4a	0.0025
e15	Bad 733 at 3a	0.0025
e16	Bad 74ls240 at u18	0.0703

5.4 PERFORMANCE ANALYSIS OF THE DIAGNOSIS STRATEGY.

The valid sets of symptoms and the values of the indices M and M' corresponding to each combination is presented in table 5.2 The definitions of these indices remain the same as that put forward in Chapter four. Assuming, predicted nature of occurrence of faults holds good, the average worst case value of M' will not deviate much from the value shown. This formulation, again, is for the first occurrence of any fault only. But assuming that probabilistic nature of occurrence of faults holds good, the value of M' for an average case will not deviate much from this.

TABLE 5.2

Valid sets of symptoms and corresponding values performance indices for start up related problems in the IBM PC.

Symptoms			M	M'
s_1	s_2	s_3		
0	0	1	11	6
0	1	0	4	1
0	1	1	15	4
1	0	0	0	0
1	0	1	12	4
1	1	0	5	2
1	1	1	16	5

Note:- If the symptom s_1 is seen then none of s_2 and s_3 can be detected at the same time. However, as seen from the AND-OR tree, the symptoms s_2 and s_3 can occur independently. This is a consequence of the inability to represent negation. The above analysis is based on the AND-OR graph only.

5.5 BOOLEAN FORMULATION OF CAUSE-EFFECT RELATIONSHIPS FOR FAULTS RELATED TO DRIVE READ/WRITE FAILURES IN IBM PC.

T → Failure of one or both the drives to read and/or to write.

The TOP event is related to the intermediate events by the boolean equation $T = s_1 + s_2 + s_3 + s_4 + s_5 + s_6$

s_1, s_2, s_3, s_4, s_5 & s_6 are the intermediate events and the annotations are as follows:-

s_1 → a particular drive does not read.

s_2 → none of the drives read.

s_3 → a particular drive reads but does not write.

s_4 → both drives read but none of them write.

s_5 → drives cannot be accessed.

s_6 → Computer locks.

The relationship between each of the intermediate events and the corresponding causes are presented in the following set of boolean equations. All the terms appearing on the RHS of these equations are basic events and they are annotated separately

s_1 = ee1 + ee2 + ee3 + f1 + a3 + a4 + b5 + c5 + d5

s_2 = ee1 + u18 + u19 + u20 + u21 + u22 + u23 + u24 + u25 + u26

s_3 = ee1 + ee4 + ee5 + e2 + b2 + c5

s_4 = ee1 + ee4 + u9 + u11

S_5 = ee5 + u12 + u14 + u16 + u17 + u29

S_6 = ee6 + ee7 + u3 + u6

The annotations of the basic events are as follows:-

ee1 → cable bad or loose.

ee2 → bad head or misalignment.

ee3 → bad or unformatted disk.

ee4 → write-protect switch is bad.

ee5 → analog connectors are corroded

ee6 → keyboard failure.

ee7 → bad RAM chip.

f1 → bad 74s38 at 1f

e5 → bad 221 at 5e

d5 → bad 74ls386 at 5d

c5 → bad 74ls74 at 5c

b5 → bad 311 at 5b.

a4 → bad 502 at 4a.

a3 → bad 733 at 3a.

u18 → bad 74ls240 at u18

u19 → bad 74ls191 at u19

u20 → bad MC4024 at u20

u21 → bad MC4024 at u21

u22 → bad 74ls112 at u22

u23 → bad 74ls161 at u23

u25 → bad 74ls112 at u25

u26 → bad 74ls02 at u26

e2 → bad 74ls14 at 2e

b2 → bad 74ls06 at 2b

u11 → bad 74ls175 at u11

u9 → bad 7438 at u9

u17 → bad 74ls273 at u17

u12 → bad 74lso4 at u12

u29 → bad 74ls02 at u29

u14 → bad 74ls08 at u14

u16 → bad 7438 at u16

u6 → bad 8288 at u6

u3 → bad 8088 at u3

5.6 BOOLEAN FORMULATION OF CAUSE-EFFECT RELATIONSHIPS FOR FAULTS RELATED TO KEYBOARD PRC LEMS IN THE IBM PC.

T → TOP event ≡ Key board error occurs

The Top event is related to the intermediate events s_1, s_2, s_3, s_4 and s_5 by the boolean equation $T = s_1 + s_2 + s_3 + s_4 + s_5$. The annotations for s_1, s_2, s_3, s_4 and s_5 are as follows:-

s_1 → Signal not reaching the mother board.

s_2 → No character is being generated.

s_3 → No signal getting to the data bus.

s_4 → Keyboard does not respond or prints bad characters.

s_5 → Keyboard stays in upper or lower case.

The relationship between the intermediate events and the corresponding events causing them is represented by the following set of boolean equations. All symbols other than those used to represent the aforesaid intermediate events are basic events.

$$s_1 = ee1$$

$$s_2 = ee2$$

$$s_3 = ee3 + u24 + u36$$

$$s_4 = s_1 + s_2 + s_3$$

$$s_5 = ee4 + ee5$$

The annotation for the basic events is as follows:-

ee1 → bad or loose cable.

ee2 → bad video circuitry.

ee3 → bad 8048 located inside keyboard

ee4 → sticky key.

ee5 → bad Caps lock key.

u24 → bad 74ls322 at u24 on system board

u36 → bad 8255 at u36 on system board

Note:-

(i) For the last two cases,the coding of symbols for basic events has been done in such a way that they point to the location of the fault.This is for ease of comprehension.

(ii)The locational details used for diagnostic messages are valid for IBM PC circuit boards only.

(iii)For Keyboard related problems,internal faults in the keyboard has not been considered as the keyboard is a third party product for an IBM PC and hence there are different types of keyboards available.

CHAPTER SIX

CONCLUSIONS

The Diagnosis strategy has been tested using the Fault-trees for the CANDU EMSR and for START UP related problems in an IBM personal computer.For testing the EMSR system,the simulator has been used and for testing START UP problems,an IBM PC/XT with known fault has been used.

6.1 RESULTS OF CASE STUDY- ONE AND DISCUSSION.

Faults were simulated using the simulator developed for electromechanical systems and using the principles of simulation described in Chapter two.In computing the values associated with the intermediate events,a single EMSR is considered and for computing the neutron population at different instants the entire EMSR system has been considered.For a 225 MWe CANDU nuclear reactor,fig.6.1 and 6.2 depict the change in observed values from the normal to the faulty state when the fault e_i(shaft 2 locked in bearings - fig.4.1)occurs at t = 1 sec after the scram signal has been received.The time base for this plot is provided by the simulator clock and while computing the neutron population,the outage of a single rod is considered at t =1 sec.It is to be

remembered that,outage of a single EMSR does not produce a failure for the entire system and hence the plots only serve to show qualitatively the difference between normal and faulted states.

In order to test the effectiveness of the diagnosis strategy a series of faults which produce the same combination of symptoms were simulated and diagnosis was made.In all these cases,only a single EMSR has been considered.Table 6.1 shows a listing of faults simulated,the time at which they occurred,the symptoms visible and the number of queries made by the user for the status of the basic events considered.The time here refers to the age of the system.

TABLE 6.1

Performance of diagnosis strategy for CANDU EMSR

Event.	Time(hr.)	Symptoms	Number of queries
e1	10000	s_4 only	6
e4	2000	s_4 only	5
e1	3000	s_4 only	5

The basic events tested in the first case were e1,e4,e5,e6,e7 and e8.The basic events tested in the second case were e4,e5,e6,e7,e8.The basic events tested in the third case were e1,e5,e6,e7 and e8.

It is thus seen that the formulation in chapter four holds good for the present system and the search trajectory conforms to expected ones.It is evident that the strategy is intelligent enough to select the most probable of all the possible faults and also uses the fact that once a fault has occurred,the probability of occurrence of the fault in near future is actually much lower assuming that the fault is repaired once it is detected. Thus, though the symptoms point to event e1 in the first case,as well as in the second,event e1 is not tested in the second case after its detection in the first.

The symptom table gives a listing of the intermediate events which have been detected. A further assumption in each case is that TOP event has occurred, which is off course quite obvious.

For a description of the events e1 and e4 and the intermediate event s_i, the annotations pertaining to the AND - OR tree of fig 4.3 is to be considered.

6.2. RESULTS OF CASE STUDY-TWO AND DISCUSSION

For testing the diagnosis strategy for the problems related to IBM personal computer, a IBM compatible PC/XT was chosen and only start up problems were considered The PC/XT selected was one which had a problem with power supply only. The reason behind doing this is

the fact that the fault-tree for the start up problems has been developed for a double drive IBM PC and since the mother boards for both IBM PC and PC/XT are identical,a large subset of common defects and corresponding diagnosis messages exists.The fault was restricted to power supply problems, because of the fact that most of the diagnostic messages involved component location level details for the IBM PC whereas the computers available for testing were only IBM compatible ones with different locational demarcations for the components on the respective boards.

Table 6.2 is a formulation showing the symptoms visible for the fault considered and the failure considered and the number of queries The symbols used conform to the fault-tree presented in Chapter five.

TABLE 6.2

Performance of diagnosis strategy for Start up problems of the IBM PC.

Event	Symptoms	Number of queries.
e_1	s_1	0

Note:-

e_1 → Power supply failure.

s_1 → Power light on and no response.

It is to be noted that fault-trees were also considered for other problems associated with the IBM personal computer viz.problems related to read/write failures,keyboard problems for the IBM personal computer.However,since most of these are third party products,the locational details of the components on their cards vary greatly but the fault-tree remains more or less the same.So the same database can be used for other systems by altering the diagnostic messages.Furthermore,detection of component level faults requires using the developed diagnosis strategy requires testing of faulty components during the search and locational details are very useful for this purpose.

6.3.SUGGESTED METHODS OF IMPROVEMENT AND SCOPE FOR FURTHER WORK.

The principal difficulty encountered in applying the diagnosis strategy in certain cases is the lack of proper reliability data.For example,while applying diagnosis strategy for fault diagnosis in electronic systems,the interface asks only for the components associated with a particular fault and computes the

equivalent failure rate using this in an oversimplified manner assuming one component for each family in the composition of the subsystem associated with the fault unless otherwise stated by the user. Rarely is failure rate data available for all the basic events associated with any fault-tree.

This problem can be tackled in two ways. In the first case, the reliability data can be obtained by rigorously computing the failure rate associated with each basic event and then entering the failure rate directly instead of the present way of representation of the reliability data. For this purpose synthetic tree model as followed in the fault-tree construction code DRAFT[6] can be applied and and component failure transfer functions may be used.

Another alternative is knowledge modification using learning methods. Heuristically, one can argue that if one particular component fails often and it is not tested while considering the most probable faults, then obviously its probability of occurrence is under projected and if it fails less often but is considered as a probable fault often, then its probability of occurrence is over projected. Thus individual a priori probability for all basic events $E_i \in E$ may be assumed to have a Coefficient C_i which can be adjusted suitably e.g if the fault occurs often but it is not included as in the set of probable faults, then the value of the Coefficient associated with its a priori probability should be incremented and

vice versa. This approach has been followed in the case of Chess playing programs. The suitable equilibrium values and equilibrium states will involve quantification of the search and once equilibrium is reached, further modification of the value of a particular component may be stopped.

Extension of the present work may be done in the following directions involving diagnosis on both simulated and actual systems: -

(i) Modification of the search strategy for tackling AND terms.

(ii) Development of a simulator for periodic forcing functions based on the present formulations.

(iii) Development of a parallel computer simulation algorithm for systems involving both feedback and feed-forward links.

(iv) Development of a multi computer on line supervisory system for on line fault monitoring in large systems.

The present search strategy cannot tackle the AND terms efficiently. Modification is contemplated in this direction. And terms will require some extra predicates for their representation.

It is evident that if each 's' in the transfer function of a system is replaced by '$j\omega$', then the frequency domain transfer

function is obtained(assuming purely sinusoidal functions). Treating each transmittance term as a complex quantity rather than a real number, the present simulation algorithm can be extended to sinusoidal periodic forcing functions of known frequencies.

Since feedback and feed-forward path computations can be done independently, a parallel algorithm can be formulated for fast simulation in cases where the system transmittances are time variant ones.

For detection of faults in a large system, fast on-line monitoring can be done by using deductive and probabilistic analysis by a number of computers sharing information each being entrusted with a particular job.

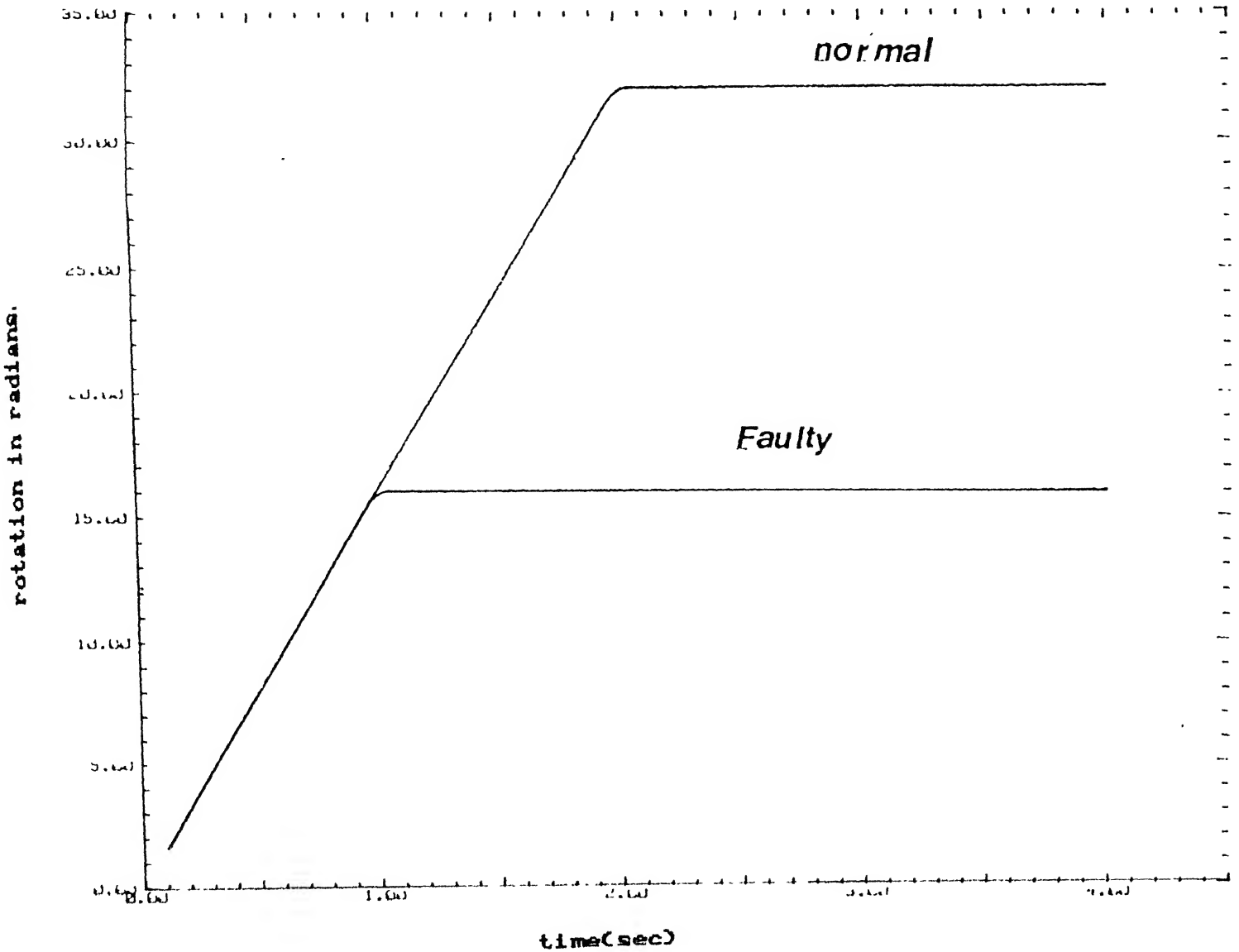


Fig.6.2 Plots showing angular displacement of shaft 2 during shut down for normal and faulty conditions

Note:- Fig. 4.1 is to be referred here.

REFERENCES

1 Kauffman, A.,Grouchko, D. and Curon, C.,'Mathematical models for study of reliability of systems',ACADEMIC PRESS,1973,2 - 12.

2 Meyers, Richard L., Wong Kam L.and Gordy, M.,'Reliability Engineering for Electronic systems',JOHN WILEY AND SONS Inc,1979.

3 Shotrliffe, E.H., 'Computer based Medical Consultations:MYCIN', Elsevier,New York,1976.

4.Rich, Elaine,'Artificial Intelligence',Mc GRAW HILL, 176 - 198

5.Sharma, D., 'Fault tree analysis of CANDU shutdown system',M-Tec thesis,Indian Institute of Technology, Kanpur (1979) 39 - 51

6.Fussell, J.B.(Ed.), 'Nuclear System Reliability and Ris Assessment',SIAM(1977)

7.Vesely, W.E., 'Analysis of Fault Trees by Kinetic Tree Theory' I 1330, Oct.1969.

8.Davis, R.,'Diagnostic reasoning based on structure and behavior' Artificial Intelligence, 24(1984) 347 - 410

9.de Kleer, J. and Williams, B.C., 'Diagnosing Multiple Faults' Artificial Intelligence, 32(1987) 97 - 130

10.Reiter, R.,'A theory of Diagnosis from First Principles Artificial Intelligence, 32(1987) 57 - 95

11.Personal Communication with Mr.A.C.Trivedi, PC Maintenan cell,Indian Institute of Technology,Kanpur.

ERRATA

1. Page 3, Section 1.2, line 4 - 'exist' should be read as 'exists'

2. Page 4, Section 1.3, last line - 'fault tree' should be read as 'AND-OR search tree'

3. Page 7, line 6 - 'dedicate' should be read as 'dedicated'.

4. Page 11, line 2- 'X(t) = $U_{-2}(t)$' should be read as 'X(t)=$U_{-1}(t)$'

5. Page 13, line 9 - 'P' should be read as 'P_2'

6. Page 15, line 1 - 'associated at' should be read as'associated with'.

7. Page 22, section 2.4, line 16-'fourth node represents' should be read as 'fifth and sixth represent'.

8. Page 23, line 19- 'cot rollers' should be read as 'controllers'.

9. Page 26, section 2.6, line 20 - 'this' should be read as 'these'.

10 Page 35, line 10 - S_k should be read as S_{k_p}.

APPENDIX - I

DEVELOPMENT OF CERTAINTY FACTOR FROM ORIGINAL MYCIN MODEL.

According to **MYCIN** (a medical diagnosis program) a hypothesis that a particular disease has occurred when certain evidence is seen is associated with two measures,a measure of belief and a measure of disbelief denoted by MB[h,e] and MD[h,e] respectively where 'e' is any evidence and 'h' is any hypothesis which can be drawn from 'e' with a certain probability.The belief and disbelief measures are defined as:-

$$MB[h,e] = 1 \text{ if } P(h) = 1$$

$$\text{else } MB[h,e] = \frac{\max[P(h|e),P(h)] - P(h)}{\max[1,0] - P(h)}$$

$$\text{and } MD[h,e] = 1 \text{ if } P(h) = 0$$

$$\text{else } MD[h,e] = \frac{\min[P(h|e),P(h)] - P(h)}{\min[1,0] - P(h)}$$

where $P(h|e)$ is the probability that the hypothesis 'h' is true given evidence 'e'.

The first term states that if the a priori probability that 'h' is 1(in which case it is known to occur)or the evidences found in its favor indicate a probability higher than its a priori probability,then there is a reason to believe that the hypothesis 'h' is true.

The second term states that if 'h' is known never to occur or if the evidence found in its favor indicate a probability lower than

the a priori probability of 'h', then there is reason to believe that 'h' is not true.

Thus if P(h|e) > P(h) then MB[h,e] > 0 & MD[h,e] = 0. Likewise, if P(h|e) < P(h), then MB[h,e] = 0 & MD[h,e] > 0.

On the basis of these measures of belief and disbelief, a certainty factor(C.F) is defined as follows:-

$$C.F = MB[h,e] - MD[h,e]$$

Thus, C.F > 0 if P(h|e) > P(h);

C.F < 0 if P(h|e) < P(h);

C.F = 0 if P(h|e) = P(h).

If the C.F for a particular hypothesis is greater than 0, then the hypothesis is accepted according to the MYCIN model.

For the present case, ,since negations are not involved, a particular symptom can only point to a fault with a certain probability only and as such only the first term(i.e MB[h,e])need be computed. The measure of disbelief need not be computed in this case and eliminations are based on deductive reasoning only.

The term P(h) in the original MYCIN model denotes the a priori probability of the hypothesis to be true and there is a qualitative relationship between the a priori probability of a particular hypothesis to be true and the probability of it to be true as suggested by the evidences in the sense that a particular hypothesis having a higher a priori probability associated with it

will have a sizable probability suggested by the evidences,when it occurs.However,in the present case,for a set of symptoms, a fault may only be considered as a probable one or its state may be deduced with certainty and the symptoms neither have any quantitative nor any qualitative relationship with the actual a priori probability of occurrence of the fault.They merely define the fault as a member or a non member of the set of probable faults.The term P(h) in the original expression for measure of belief is replaced by a probability $P(e_i) = 1/n$ where n is the number of elements in the set of probable faults obtained after the first tier of search based on deductive reasoning.The probability term $P(e_i)$,therefore assumes that all the faults in the aforesaid faults are equally probable and denotes an average value.

The term P(h|e) can be replaced by the term $P(e_i|T)$ to indicate the probability of occurrence of a fault(i.e a basic event) e_i relative to other basic events in the set of probable faults E_v when the TOP event occurs, (not known definitely) when the TOP event occurs and is accompanied by the occurrence of a set of $symptoms_i$. The term $P(T|e_i)$ is computed in the following manner.

Let $P(e_i)_a = 1 - \exp(-\lambda_i T_i)$ denote the actual probability of occurrence of a fault where λ_i denotes the equivalent failure rate associated with the fault(basic event) e_i and T_i is the time of operation for the components assuming the components to be fresh.It is to be noted that once a fault is detected and the faulty components are replaced or repaired,they are supposed to start a fresh life.

After computing the value of $P(e_i)_a$, the next step is to see how it compares with the corresponding probability values for the other

probable faults. This is achieved through normalization and $P(e_i|T)$ then indicates the probability value associated with the fault e_i relative to others in the set of probable faults. Normalized probability $P(e_i|T)$ can be obtained using the relation below: -

$$P(e_i|T) = P(e_i)_a / \left(\sum_{j=1}^{n} P(e_j)_a \right)$$

and the corresponding certainty factor is defined as

$$C.F(e_i|T) = P(e_i|T) - P(e_i)$$

All faults with $C.F \geq 0$ are classed as more probable faults and faults with $C.F \leq 0$ are less probable faults. It is evident that the more probable faults have a relative probability more than the average value and the less probable faults have a value less than the average.

From the discussions made so far two things become evident. These are: -

(i) In a set of all probable faults no matter what the individual a priori probability values for the probable faults be, only the most probable ones amongst those are selected.

(ii) As the time of operation associated with a fault increases, without the fault occurring, the a priori probability of that fault increases. This indicates, that for the same set of symptoms, the certainty factor associated with the fault increases with time. In other words, the fault becomes increasingly likely for the same set of symptoms.

RUNNING THE SOFTWARE

A.2.1. RUNNING THE SIMULATOR.

To run the simulator one has to run the SIMUL.EXE file. This is the stand alone version of the simulator software and can be used to simulate electromechanical systems. It is a menu-driven software and the main menu permits the following modes:-

(1). Build

(2). Edit

(3). Run

(4). Fault

(5). Quit

Selection in the main-menu is done by hitting the space bar(this places the selector at the appropriate choice) and pressing the letter 's' or 'S'. Pressing the space-bar after the selector is placed at the 5^{th} menu item(i.e Quit) resets the position of the selector to the position of the first menu item i.e Build.

Each item in the main menu has a sub-menu associated with it. The Build sub-menu consists of the following items:-

(a) Build a new system

(b) Import an already created system

Using the first option, an electromechanical system can be built. The simulator provides an extremely user friendly interface for building a system. As soon as the Build option is selected, four

windows appear on the screen.The largest of these four is the canvas.The extreme right window shows the menu of the forward path components(series) that can be simulated.The bottom-most window is the dialogue window and it guides the user during the building process.It also takes inputs from the user in the graphics mode(i.e during building).The selection of an appropriate location on the canvas is done by moving the location selector(a small white circle) in the canvas.The selector movement commands are, 'h' for moving left,'j' for moving downwards,'k' for moving upwards and 'l' for moving right.To build a particular component,one has to take the locator to an appropriate location on the canvas, press any key other than the aforesaid four to inhibit the locator movement routines and then press the appropriate key as directed by the menu window.This builds the sub graph corresponding to the item chosen.The user is prompted to type in the values of the required parameters(like armature resistance for a motor or gear ratio for a gear) by the dialogue window.It is to be noted that in the building mode, all the parameters(be real or integer) are read in as a 8 character string and is converted to an integer or real as required by the particular case.The next step is numbering of the items built.This process establishes the connectivity between the individual sub graphs and builds the signal flow graph for the above system.This is again, to be fed in by the user through the dialogue window and each is a two character string.The numbering routine automatically indicates the number of which component it wants.

The build routine then asks the user whether feed-back/feed-forward paths need be built.The answer is 'y' for

yes and 'n' for no.It may be mentioned here that all the feed-back and feed-forward paths within any particular series path component are tackled when that component is built.If the user responds with an 'yes' to the above query,the menu window shows him which key to press for beginning a feed-back/feed-forward path and which key to press to terminate it.To initiate a feed back from one particular component,the user has to take the locator to the appropriate component,place it in within the circular zone marked there,inhibit the locator movement and press 'b' or 'B'.To terminate the feed-back he has to take the locator to the appropriate component and follow it up with identical actions except that he has to press the key 's' or 'S' to select the termination point.During the process of selection(both during feeding in and feeding out) the message window(right down) shows which component is being selected and the dialogue window states what can be fed out or fed in.If a valid feed-back or feed forward path is formed,the dialogue window asks for the value of the path transmittance associated with it.Invalid paths are automatically taken care of.At this stage after entering the path transmittance if the user wishes to come out of this path building routine he simply press 'q' to terminate the process,anything else returns the control to the path builder routine.This routine is followed by two other routines which are invoked if the user so desires viz. the routine for building timer relays and associating it with contactors and the routine involving the nuclear interface for the simulator.All these routines are guided by detailed instructions through the dialogue window and is therefore left out for the sake of brevity.The locator movement and the selection routines remain the same in all cases.

The most important part of the building process is the routine involving templating. The user is first asked to enter the name of the file where this template details will be stored. The user has to start first with the intermediate events and select the appropriate channel(a particular node in the graph) and mention whether he wants instrumentation for that or not. If he does require an instrumentation, it is recorded, compared with the normal value and messages are sent. The user has to repeat the same process for the basic events and for the TOP event. It may be noted that for ascertaining the occurrence of the an intermediate event nodal values are cobsidered and for ascertaining the occurrence of basic events, the corresponding branch transmittances are considered.

After the construction part is over, the user is asked if he wants to save the system. If he does, he has to enter the name of the dosfile(with full drive and directory specification) without any extension and the system details are stored there.

The build option also permits the user to import an already build system in the work area. In order to import the system, the user again has to type in the filename in the same way as he does while saving.

The Edit option of the main menu permits the user to edit an imported system. This option can be used to edit series component, feed-back/feed-forward components , timer relays and attributes associated with the nuclear interface. The process is done in the text mode and the user is provided with instructions at every stage. The principal edit codes available are 'a' meaning append, 'd' meaning delete, 'c' meaning change and 'q' meaning quit edit mode for that particular group of components.

The **Run** option can be selected for a normal case and also for a an abnormal case.For normal case, the Run option simulates the system response(after linearization,when required) displays the results of some selected nodes(to be selected by the user).It also keeps record of the normal values associated with all the intermediate events and the TOP event.These values are stored in a file designated by the user.The user has to input the initial and final values of the input signal,the type of function he wants, the time-step required for computation and the total time of run.The timer routine associated with the Run module keeps track of the simulated time and displays it.

The **Fault** option involves several sub options involving the creation,import,saving,editing and running of the faulted systems.A library of routines for these faults have been created covering all the series and feed-back, feed-forward path and nuclear interface components and the user has to specify the code(a two character string) for the fault concerned, the particular component, and the time(w.r.t the simulator clock) at which he wants the fault to occur.The routines involving building,editing,saving and retrieving of faulted system details are supported by detailed instructions.However,the Run module for fault mode involves special consideration.It now not only computes the system response, but also compares the values with the normal ones and generates a report(report filename to be provided by the user) stating which are the nodes, and/or branches at which it has found changes.

The simulator output has now got to be interfaced with the diagnosis softaware.This is done by a batch file 'SETTUP' which contains a number of files(2 actually) which do the decision

making, match the changed parameters with the fault template for the system and tell the user which of the intermediate events(having instrumentation) have occurred.It also maintains a 'teller' window which can be accessed by the user to know the status of any other basic or intermediate event.The format of setting up the interface is 'Settup<*rport filename*> <*template filename*>.

A.2.2.BUILDING THE KNOWLEDGE BASE FOR DIAGNOSIS.

This involves telling the diagnosis software what are the intermediate events,what are the basic events,what are the failure rates associated with each component or what are the components associated with each basic event and what are the causal relation ships between the intermediate events and the corresponding basic events.This is performed in steps and each step is accompanied by detailed instructions.For calculating the equivalent failure rates associated with each basic event the failure rate data of some of the components is incorporated in a database('MAST.DAT') which must remain in the same directory as the interface software.The database has been prepared using [2] and [5].For preparing the interface,the command format is 'INTERFC<*filename* > where *filename* is the name of the family of files which store the knowledge base.The knowledge base is stored in 6 files of the form *filename*.*.

A.2.3. RUNNING THE DIAGNOSIS SOFTWARE.

For running the diagnosis software, the batchfile DIAG is to be invoked. The format is DIAG<*filename*> where *filename* is the same as that used while interfacing. The diagnosis software is based on the algorithm outlined in chapter three and is accomplished in steps. It also records the components which are being replaced and at what time for future reference. The process is highly user interactive and every step is provided with accompanying messages.

Originally it was contemplated to run the simulator, the setting up software and the diagnosis software on the same computer using standard software packages like MS Windows. However, this is not being done for the present version of the software and the entire software has to run on two computers, the setting up software on one and the diagnosis on the other. Bridging of the two has not been done.

NETP-1991-M-GUP-EXP